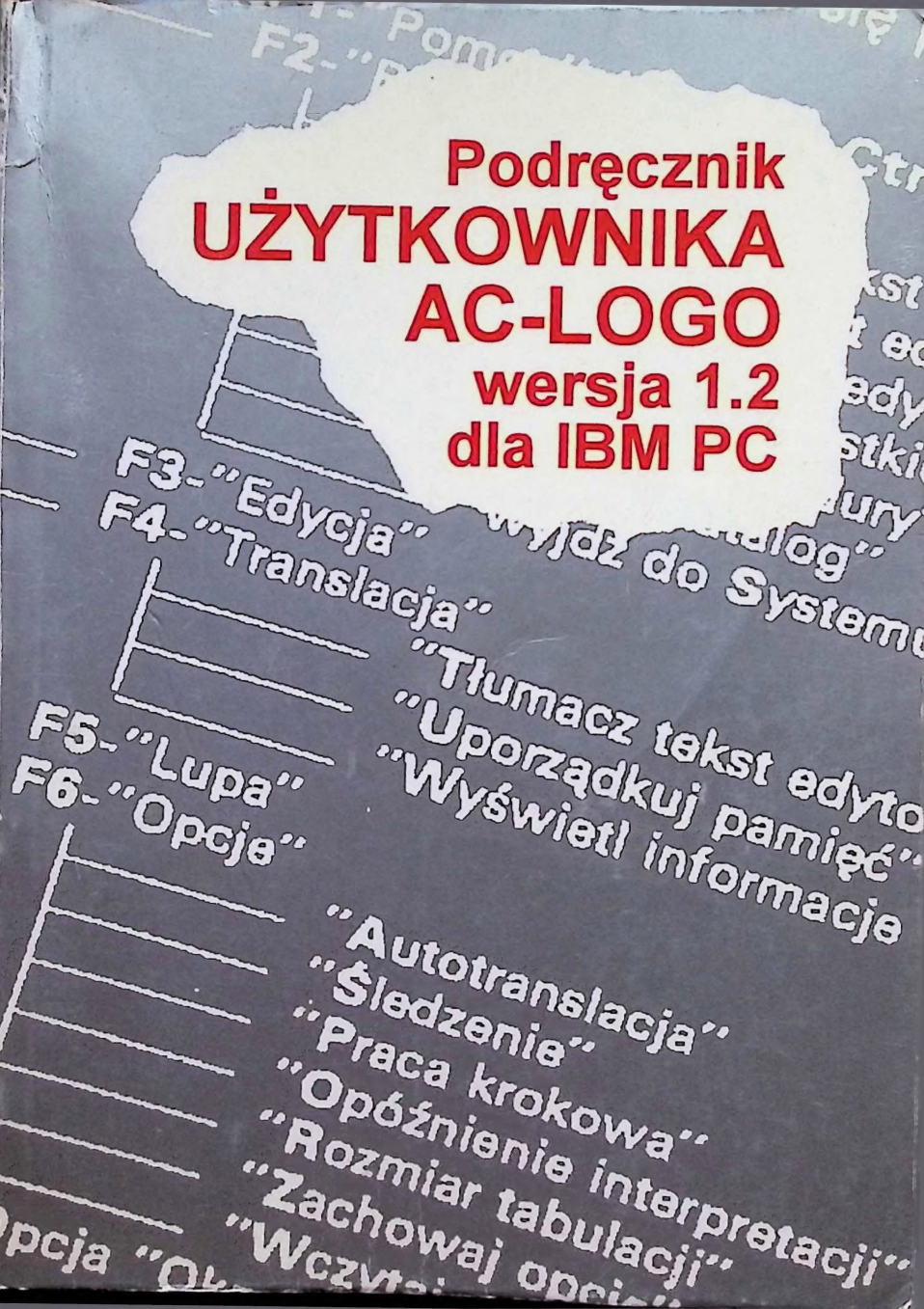

Podręcznik
UŻYTKOWNIKA
AC-LOGO
wersja 1.2
dla IBM PC
F3-"Edycja"
F4-"Translacja"
"Tłumacz tekst edyto
"Uporządkuj pamięć"
F5-"Lupa"
"Wyświetl informacje
F6-"Opcja"
"Autotranslacja"
"Śledzenie"
"Praca krokowa"
"Opóźnienie interpretacji"
"Rozmiar tabulacji"
"Zachowaj opcje"

Podręcznik
UŻYTKOWNIKA AC-LOGO
Wersja 1.2 dla IBM PC

Podręcznik użytkownika AC-LOGO

Wersja 1.2 dla IBM PC

Jelenia Góra 1994

Na zlecenie Ministerstwa Edukacji Narodowej
system i dokumentację opracowała Firma AdvaCom
(Przedsiębiorstwo Innowacyjno-Wdrożeniowe) Spółka z o.o. w Poznaniu

Projekt okładki i redakcja techniczna
ROMUALD WITCZAK

© Copyright AdvaCom Spółka z o.o. Poznań

© Copyright Ministerstwo Edukacji Narodowej

ISBN 83-903197-0-05

Wydawca instrukcji i dystrybucja systemu AC-LOGO — Wydawnictwo OFEK Jelenia Góra
ul. 1 Maja 43, tel. 255-41

Druk i oprawa: Wrocławska Drukarnia Naukowa

SPIS TREŚCI

1. WSTĘP 7

2. POJĘCIA PODSTAWOWE 9
2. 1. Elementy obrazu 9
2. 2. Ekran 9
2. 3. Tryb tekstowy i tryb graficzny 10
2. 4. Okienko graficzne 13

3. ELEMENTY JĘZYKA 16
3. 1. Znaki 16
3. 2. Separatory 17
3. 3. Komentarze 17
3. 4. Typy danych 18
3. 5. Wyrażenia 21
3. 6. Procedury 26
3. 7. Funkcje 30
3. 8. Procedury i funkcje pierwotne 30
3. 9. Zmienne 31
3.10. Przekazywanie parametrów do procedur 32

4. SYSTEM PROGRAMOWANIA AC-LOGO 37
4. 1. Programy tłumaczące 41

5. EDYTOR WIERSZOWY 43

6. EDYTOR EKRANOWY 45
6. 1. Elementy informacyjne okna edytora 46
6. 2. Kursory 46
6. 3. Przesuwanie kursora tekstowego 47
6. 4. Kasowanie tekstu 49
6. 5. Operacje na bloku tekstu 50
6. 6. Przeszukiwanie i zamiana tekstu 52
6. 7. Inne operacje 55
6. 8. Zakończenie edycji 56

7. SYSTEM OKIENKOWY 57
7. 1. Ogólne zasady poruszania się w systemie menu 57
7. 2. Opis opcji głównego menu 59
7. 3. System podpowiedzi 72
7. 4. Opis edytora wierszowego dla systemu menu 74
7. 5. Opis klawiszy specjalnych 74

8. LEKSYKON PROCEDUR PIERWOTNYCH 77

Dodatki
A. Rozszerzony kod ASCII LATIN-II 227
B. Rozmieszczenie znaków na klawiaturze 228
C. Słowniczek polsko-angielski nazw procedur pierwotnych 231
D. Słowniczek angielsko-polski nazw procedur pierwotnych 238
E. Spis komunikatów błędów 245
F. Opis pliku konfiguracyjnego 253
G. Opis formatów plików AC-LOGO 262
H. Śledzenie ruchu żółwia z poziomu innych programów 266

Literatura 268

1. WSTĘP

U podstaw języka LOGO leżą odkrycia współczesnej psychologii, a w szczególności badania J. Piageta. Potwierdziły one szczególną rolę procesu aktywnego uczenia w procesie kształtowania się twórczej osobowości ucznia. Proces ten polega na samodzielnym budowaniu modeli otaczającego ucznia świata i konfrontowaniu ich z rzeczywistością. LOGO jest narzędziem ułatwiającym uczniowi samodzielne konstruowanie tych modeli i badanie praw rządzących nimi. Programowanie w LOGO polega na stopniowym definiowaniu znaczenia kolejnych słów w języku zbliżonym do naturalnego. Słowa te są nazwami procedur, których treść określa ich znaczenie. Takie podejście umożliwia dopasowywanie struktury programu do struktury problemu, a nie odwrotnie. Rozwiązanie problemu polega tu na dzieleniu go na kolejne podproblemy i określaniu związków między nimi. Podproblemy te są następnie niezależnie rozwiązywane w taki sam sposób, aż do najniższego poziomu, na którym znajdują się podproblemy rozwiązane przez procedury pierwotne języka LOGO.

Język programowania LOGO ma również liczne zalety informatyczne. Jest to język o strukturach danych i mechanizmach języków sztucznej inteligencji, dzięki czemu można wyrażać w nim złożone problemy w sposób bardzo zwięzły i prosty, przy zachowaniu dużej czytelności. Ponadto, jest to język interakcyjny, w którym efekty działań oraz popełniane błędy są widoczne natychmiast i natychmiast mogą być korygowane.

Wprowadzanie LOGO do szkół w najbardziej rozwiniętych krajach świata rozpoczęto w latach osiemdziesiątych. Również w Polsce istniała świadomość zalet tego języka i celowości udostępnienia go polskim uczniom. Wprowadzenie LOGO do polskich szkół wiązało się z koniecznością przetłumaczenia jego słownictwa na język polski. Jest to konieczne dla osiągnięcia wymaganej komunikatywności niezbędnej dla spełnienia założonych celów dydaktycznych. W połowie lat osiemdziesiątych powstała polska wersja interpretera LOGO dla mikrokomputera Sinclair Spectrum. Kolejnym istotnym etapem wprowadzania języka LOGO do polskich szkół było opracowanie przez pracowników Instytutu Informatyki Politechniki Poznań-

skiej kompilatora o nazwie "Junior LOGO", przeznaczonego dla mikrokomputerów edukacyjnych Elwro 800 Junior. Jednakże w związku z coraz powszechniejszym występowaniem w szkołach mikrokomputerów klasy IBM PC niezbędne stało się opracowanie polskiej wersji języka LOGO dla tych mikrokomputerów. Po około dwóch latach intensywnych prac powstał zintegrowany system programowania w języku LOGO, o nazwie AC-LOGO, przeznaczony dla mikrokomputerów klasy IBM PC. Podstawowe cechy tego systemu to:

- nowoczesny, przejrzysty interfejs z użytkownikiem oparty na systemie okien graficznych i tekstowych;
- duża szybkość działania;
- duża liczba procedur pierwotnych;
- wygodny w użyciu edytor ekranowy, o bogatym zestawie poleceń i standardowej mnemonice;
- możliwość stosowania komentarzy oraz dowolnego formatowania tekstu procedur, w celu strukturalnego zapisu algorytmu;
- możliwość sterowania prędkością pracy i śledzenia wykonywania procedur na poziomie programu źródłowego;
- rozbudowany, kontekstowy i interakcyjny system pomocy (HELP).

Niniejszy podręcznik użytkownika zawiera pełen zestaw szczegółowych informacji niezbędnych do posługiwania się systemem programowania AC-LOGO oraz dialektem AC-LOGO.

Podręcznik rozpoczynają definicje i wyjaśnienia podstawowych pojęć dotyczących grafiki komputerowej (rozdział 2) oraz opisy elementów języka LOGO: znaków, separatorów, komentarzy, typów danych, wyrażeń, procedur, funkcji, procedur pierwotnych i zmiennych (rozdział 3).

W rozdziałach 4, 5 i 6 opisano sposób posługiwania się kompilatorem i edytorami — wierszowym i ekranowym.

W rozdziale 7 opisano zasady posługiwania się systemem okienkowym.

Rozdział 8 składa się z opisów procedur pierwotnych LOGO ułożonych alfabetycznie, bogato ilustrowanych prostymi przykładami. Opisy te pozwalają na pełne zrozumienie funkcjonowania każdej procedury pierwotnej.

Książkę kończą dodatki, w których zawarto tablicę rozszerzonego kodu ASCII w wersji LATIN-II używanego w mikrokomputerach typu IBM PC, rozmieszczenie znaków na klawiaturze tego komputera, dwa słowniczki nazw procedur pierwotnych: polsko-angielski i angielsko-polski, spis komunikatów błędów, opis poleceń pliku konfiguracyjnego, opis formatów plików z rysunkami i generatorami znaków oraz sposób śledzenia ruchów żółwia przez rezydentne programy systemu operacyjnego MS-DOS.

2. POJĘCIA PODSTAWOWE

2.1. Elementy obrazu

Punkt jest elementarną jednostką obrazu komputerowego. Kombinacje punktów tworzą znaki (litery, cyfry, znaki interpunkcyjne itp.) oraz figury geometryczne (proste, kwadraty, okręgi, koła itp.). Tworzenie obrazu polega na nadawaniu odpowiednich kolorów poszczególnym punktom.

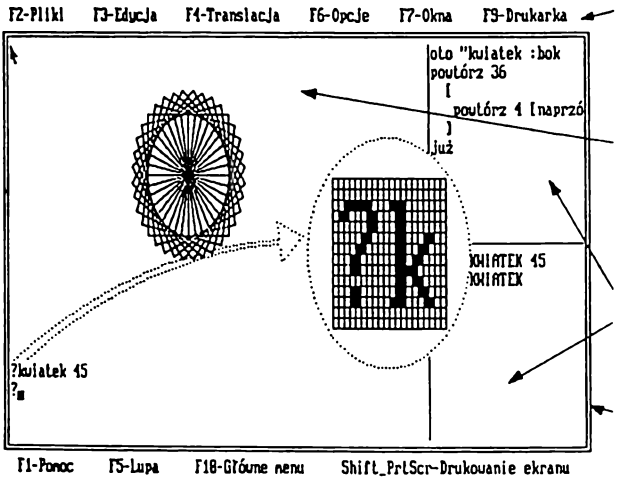
Znak jest reprezentowany na ekranie monitora przez prostokątną matrycę punktów — tak zwane pole znakowe. Ciąg znaków tworzy tekst.

W AC-LOGO znaki są wyświetlane za pomocą generatorów znaków. **Generator znaków** to plik, w którym są pamiętane wzory dla wszystkich znaków z jednej grupy, przy czym znaki te mają jednakową wielkość (rozmiary pola znakowego w kierunku pionowym i poziomym). Nazwy wczytywanych generatorów znaków są pamiętane w pliku konfiguracyjnym. Jednocześnie można wczytać maksymalnie 16 różnych generatorów, przy czym każdy z nich może zawierać do 256 różnych wzorów znaków. Zawsze muszą być wczytane dwa podstawowe generatory znaków, o matrycy 8 na 14 punktów: SYSTEM i LAT8X14. Pozostałe generatory znaków są wczytywane opcjonalnie. Należy pamiętać, że wczytanie zbyt dużej liczby generatorów znaków jednocześnie powoduje ograniczenie dostępnej dla programów LOGO pamięci komputera. Zmiana aktualnie wybranego generatora znaków następuje po wykonaniu procedury pierwotnej .GENERATOR (opisanej w rozdziale 8).

2.2. Ekran

Na rysunku 2.1 przedstawiono elementy obrazu monitora mikrokomputera IBM PC z uruchomionym systemem AC-LOGO. Wyróżniamy w nim okno tekstowo-graficzne, okienka śledzenia, menu okienkowe i ramkę. Ponadto na ekranie mogą się pojawiać okienka systemu okienkowego.

Obecność na ekranie okienek śledzenia oraz menu okienkowego jest opcjonalna. Wielkość okna tekstowo-graficznego zależy od posiadanej karty graficznej i wynosi maksymalnie 704 punkty w poziomie i 322 punkty w pionie.



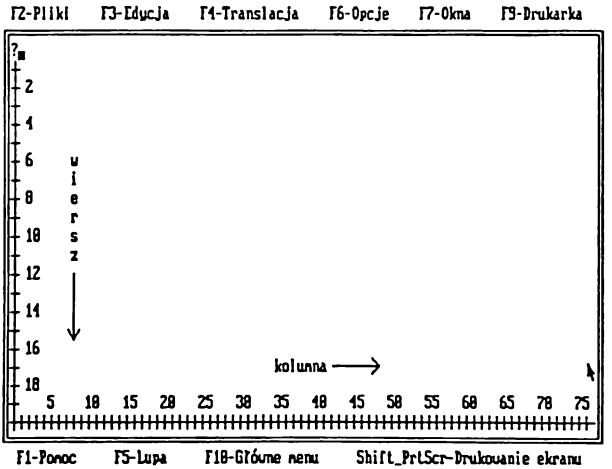
Rys. 2.1. Elementy ekranu komputera pracującego pod kontrolą AC-LOGO

Punktom i znakom z wnętrza okienek oraz ramce każdego okienka można nadawać kolory. Możliwość nadawania kolorów zależy od posiadanej karty graficznej — dla karty monochromatycznej każdy punkt może być wyświetlony w jednym z dwóch kolorów (umownie nazwanych "czarnym" i "białym"), karta kolorowa udostępnia jednocześnie do 16 kolorów. W odróżnieniu od kompilatora Junior-LOGO, nie przewiduje się wprowadzenia atrybutów punktu: podwójnej jasności (zastąpionego przez zwiększoną liczbę dostępnych kolorów) i migotania. Sposób nadawania kolorów wyjaśniono dokładniej w punkcie 2.5.

2.3. Tryb tekstowy i tryb graficzny

Okno graficzno-tekstowe AC-LOGO może działać w dwóch trybach — tekstowym i graficznym. W trybie tekstowym w całym oknie są umieszcza-

ne znaki. Pozycję, w której na ekranie jest umieszczany kolejny znak, wskazuje kursor tekstowy. Jest nim jasny prostokąt o wysokości równej wysokości znaku lub jej połowie (w zależności od trybu wprowadzania znaków wyjaśnionego później). Pozycję kursora można odczytywać oraz zmieniać. Jest ona pamiętana w konwencji: numer kolumny, numer wiersza. Pozycją kursora o współrzędnych (0,0) jest lewy górny róg okna (por. rysunek 2.2). Rozmiary okna w trybie tekstowym zależą od posiadanej karty graficznej, opcji "LUPA" i rozmiaru okienek śledzenia (por. rozdział 3).



Rys. 2.2. Współrzędne kursora tekstowego

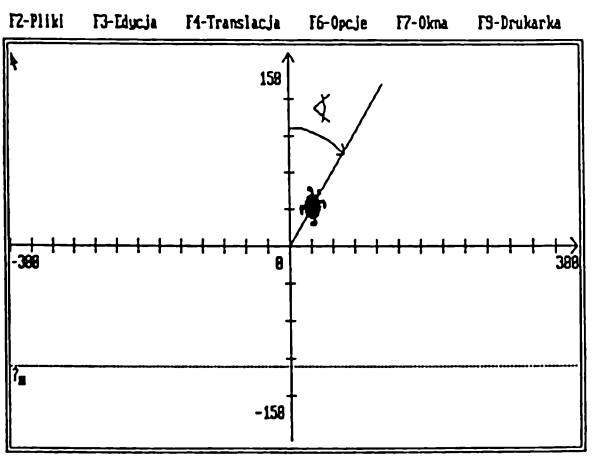
Górna część okna tekstowo-graficznego może być przeznaczona do wyświetlania grafiki. Podział okna na część tekstową i graficzną można zmieniać za pomocą procedury pierwotnej DZIELEKRAN (ang. SETSPLIT; por. rozdział 8). Standardowo liczba wierszy tekstu w oknie tekstowo-graficznym wynosi 4.

W AC-LOGO współrzędne kursora nie są związane z częścią "tekstową" okna tekstowo-graficznego, lecz z całym tym oknem. Oznacza to, że można ustawić kursor w dowolnym miejscu okna. Istnieje zatem możliwość umieszczania napisów także w części "graficznej" okna.

W oknie graficzno-tekstowym można nadawać kolory tła i atramentu poszczególnym punktom części "graficznej" tego okna. Pozycję punktu, którego stan można zmieniać w danym momencie, wskazuje kursor graficzny

zwany żółwiem. Pozycję żółwia można odczytywać i zmieniać. Jest ona pamiętana jako para liczb z zakresu od −32000 do 32000, gdzie pierwsza liczba jest współrzędną x, a druga y w prostokątnym, kartezjańskim układzie współrzędnych o początku znajdującym się w centralnym punkcie ekranu (por. rysunek 2.3). Żółw może poruszać się po ekranie i poza nim. Kierunek poruszania się żółwia jest określony przez kąt jaki tworzy prosta, po której porusza się żółw, z dodatnią półosią OY układu współrzędnych. Kąt ten może być odczytywany oraz zmieniany.

Do zmiany stanu punktów ekranu służy pisak żółwia. Pisak ten ma przypisane atrybuty: kolor atramentu i kolor tła, które odpowiednio nadaje punktom okna tekstowo-graficznego, w obszarze którego porusza się żółw. Pisak może znajdować się w jednym z czterech stanów: "podniesienie", "opuszczenie", "ścieranie" i "odwracanie". W stanie "podniesienie" poruszający się pisak nie zmienia stanu punktów okna. W stanie "opuszczenie" każde przesunięcie żółwia powoduje nadanie punktowi, nad którym przesunięto pisak, koloru atramentu. Stan "ścieranie" różni się od stanu "opuszczony" tym, że punktowi, nad którym przesunięto pisak, nadaje się kolor tła, a nie kolor atramentu. Stan "odwracanie" różni się natomiast tym, że punkt, nad którym przesunięto pisak, przyjmuje kolor "dopełnieniowy" — na przykład kolor biały, jeśli był w kolorze czarnym itp.



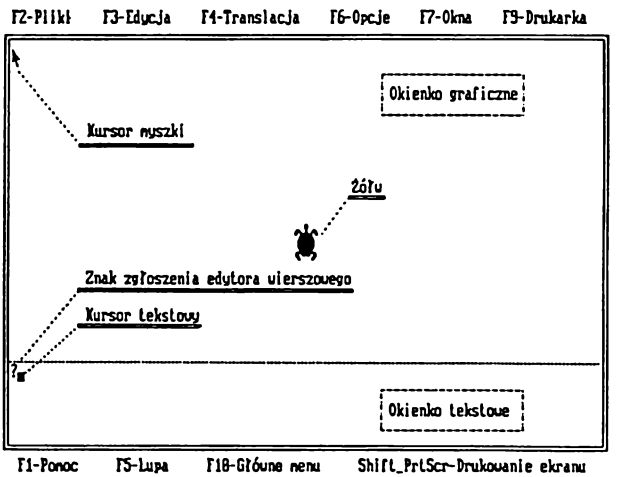
Rys. 2.3. Układ współrzędnych graficznych

2.4. Okienko graficzne

Część okna tekstowo-graficznego przeznaczoną na wykonywanie rysunków nazywa się okienkiem graficznym. Po wywołaniu systemu programowania AC-LOGO całe okno tekstowo-graficzne jest przeznaczone na teksty. Podział okna na część tekstową i graficzną następuje dopiero po wykonaniu procedury pierwotnej CZYŚĆ (ang. CLEARSCREEN) lub po wykonaniu pierwszej procedury przemieszczającej żółwia (por. rysunek 2.4). Współrzędne okienka graficznego zależą od posiadanej karty graficznej oraz rozmiarów innych okienek na ekranie.

Rozmiary okienka graficznego mogą być zmieniane w dwojaki sposób.

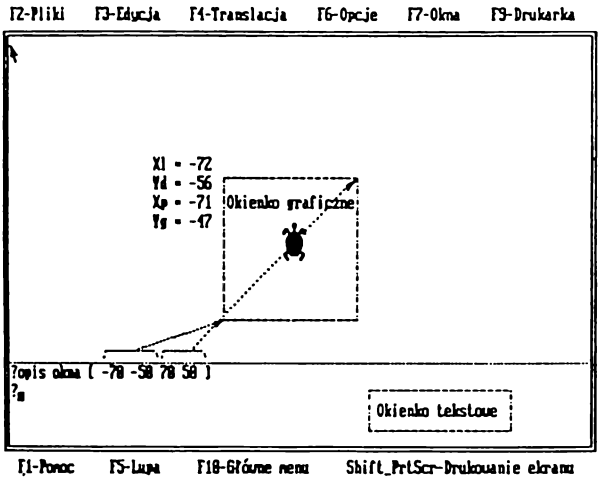
Pierwszy sposób polega na ustaleniu rozmiaru okienka tekstowego za pomocą procedury pierwotnej DZIELEKRAN (ang. SETSPLIT) i zmazaniu ekranu za pomocą procedury pierwotnej CZYŚĆ (ang. CLEARSCREEN). Okienko graficzne zajmuje wówczas część okna tekstowo-graficznego nie zajętą przez okienko tekstowe. Za pomocą tej metody można zatem ustalać jedynie współrzędną y dolnej krawędzi okienka.



Rys. 2.4. Okienko graficzne i okienko tekstowe

Drugą metodą ustalenia wymiarów okienka graficznego jest zastosowanie procedury pierwotnej OPISOKNA (ang. DEFINEWINDOW; por. rozdział 8). Możliwe jest wówczas ustalenie wszystkich wymiarów okienka.

Wymiary okienka nie mogą jednak przekraczać wymiarów części graficznej okna tekstowo-graficznego. Należy zaznaczyć, że współrzędne okienka graficznego są rozszerzane tak, aby szerokość tego okienka była wielokrotnością liczby 8 (okienko jest definiowane z dokładnością do bajtu, podobnie jak znaki — por. opis generatorów znaków powyżej).



Rys. 2.5. Deklarowanie okienka graficznego

Zdefiniowanie okienka graficznego pierwszą z opisanych metod powoduje zmazanie całej dotychczasowej zawartości okna tekstowo-graficznego. Natomiast druga metoda nie niszczy aktualnego obrazu okna, a w szczególności poprzedniego okienka graficznego.

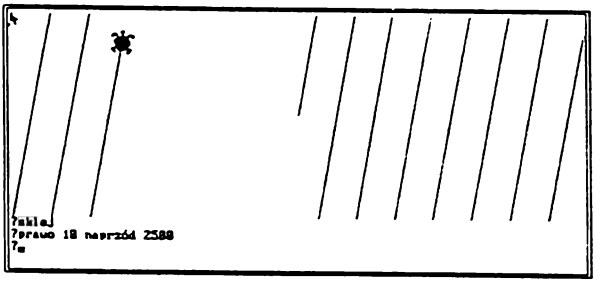
Okienko graficzne może znajdować się w jednym z trzech trybów: "sklej", "okno" i "płot".

W trybie "sklej" przeciwległe krawędzie okienka są traktowane tak, jakby były połączone, tzn. żółw po przejściu górnej krawędzi okienka graficznego pojawia się na jego dolnej krawędzi, a po przejściu prawej krawędzi pojawia się na lewej (por. rysunek 2.6a).

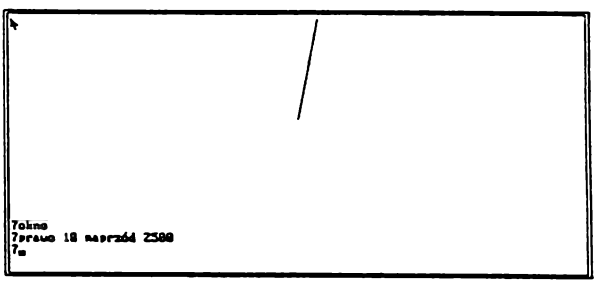
W trybie "okno" żółw po wyjściu poza krawędź okienka graficznego przestaje być widoczny, chociaż dalej może się poruszać (por. rysunek 2.6b).

W trybie "płot" wyjście żółwia poza okienko graficzne jest traktowane jako błąd (por. rysunek 2.6c).

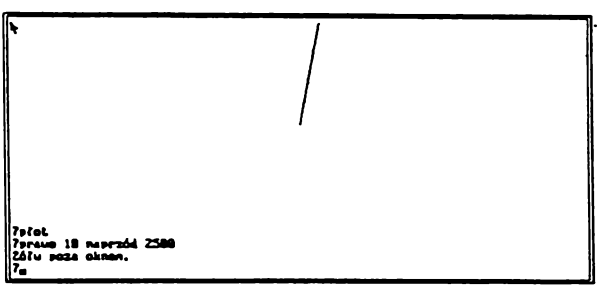
a)



b)



c)



Rys. 2.6. Efekty przesunięcia żółwia w różnych trybach okienka:
a) w trybie "sklej" b) w trybie "okno" c) w trybie "plot"

3. ELEMENTY JĘZYKA

3.1. Znaki

W programach napisanych w LOGO można używać wszystkich znaków alfabetu LATIN-II (por. Dodatek A). Dostępne są zatem następujące symbole:
— wielkie litery alfabetu polskiego od A do Ż
— małe litery alfabetu polskiego od a do ż
— wielkie i małe litery innych europejskich alfabetów narodowych
— znaki semigraficzne
— cyfry od 0 do 9
— znaki specjalne ! " # $ % & ' ( ) _ + ; * : { [}] ? / ¦ ^ ~ < , > . = − @ \ — znak spacji
— znak nowego wiersza (CR,ENTER)
— znak sygnału dźwiękowego (BELL).

Konstrukcją równoważną pojedynczemu znakowi jest dwuznak. Składa się on z dowolnego znaku poprzedzonego znakiem ukośnikiem \ (np. \a, \!, \\). Wartością dwuznaku jest jego drugi znak pozbawiony znaczenia gramatycznego. Dwuznak stosuje się np. w celu zapobieżenia interpretacji odpowiednich znaków jako separatorów.

W celu pozbawienia znaczenia gramatycznego ciągu znaków należy ująć go w apostrofy, przy czym rolę prawego apostrofu może pełnić znak nowego wiersza CR (ENTER).

Przykłady

ty i ja	— trzy słowa: ty, i, ja;
ty\ i\ ja	— jedno słowo: ty i ja;
'ty i ja'	— jedno słowo: ty i ja;
'ty i ja	— jedno słowo: ty i ja; prawy apostrof zastąpiony znakiem nowego wiersza CR;
'ty i 'ja	— jedno słowo: ty i ja;
'ty i'\ ja	— jedno słowo: ty i ja.

3.2. Separatory

Separatory służą do oddzielania poszczególnych jednostek leksykalnych języka LOGO, takich jak słowa, liczby itp. Separatorami są znaki spacji, nowego wiersza oraz nawiasy kwadratowe i okrągłe. Ciąg separatorów jest traktowany jak jeden separator. Zatem stosowanie pustych wierszy i wcięć w celu przejrzystego zapisu programu nie powoduje spowolnienia działania tego programu.

3.3. Komentarze

Dowolny ciąg znaków ujęty w nawiasy klamrowe { } i poprzedzony separatorem jest komentarzem, czyli objaśnieniem dodanym do programu. Komentarze są pomijane przez kompilator LOGO w trakcie tłumaczenia programu, nie powodują zatem spowolnienia jego działania. Wewnątrz komentarza nie może wystąpić znak }.

Przykłady

Instrukcja

```
naprzód 40 {idź o 40 kroków naprzód}
```

jest równoważna instrukcji

```
naprzód 40
```

Instrukcja

```
naprzód {czterdzieści} 40
```

jest równoważna

```
naprzód 40
```

Instrukcja

```
np 20 {kom1 {kom2} kom3} pw 90
```

jest równoważna

```
np 20 kom3} pw 90
```

ponieważ tekst kom3} jest już poza komentarzem. Zostanie on potraktowany przez kompilator jako nazwa procedury. Jeśli nazwa taka nie została uprzednio zadeklarowana, to wystąpi błąd. Ciąg znaków: Ala{Ola}Jan nie zawiera komentarza.

3.4. Typy danych

W języku LOGO wyróżnia się dane i procedury. Dane są przedmiotem, a procedury podmiotem przetwarzania. Wykonywanie programu napisanego w LOGO polega na przetwarzaniu danych przez procedury. Warto zaznaczyć, że w LOGO nie ma ścisłego rozgraniczenia pomiędzy danymi a procedurami — ten sam obiekt może być w pewnych sytuacjach przedmiotem, a w innych podmiotem przetwarzania.

W LOGO wyróżniamy dwa podstawowe typy danych: słowa i listy.

Słowa

Słowem w rozumieniu LOGO jest ciąg znaków nie zawierający separatorów[1]. Ciąg ten może być pusty.

Słowa w LOGO spełniają różne role: mogą być liczbami, wartościami logicznymi, nazwami zmiennych i procedur, elementami list lub po prostu ciągami znaków. Znaczenie słów w LOGO zależy od kontekstu.

W AC-LOGO liczba jest szczególnym słowem. Liczby występują w postaci dziesiętnej lub szesnastkowej. Znaczenie liczby zależy od kontekstu jeżeli jest ona użyta w wyrażeniu arytmetycznym, to jest traktowana jako wartość liczbowa, natomiast w pozostałych przypadkach jest traktowana tak jak inne słowa.

Liczbami dziesiętnymi są ciągi znaków postaci:

*sm.n*E*vw*

lub:

sm.nevw

gdzie:

s oznacza znak liczby + lub − ; jeśli nie występuje, to znakiem liczby jest + ;

---

[1] Ze względów pragmatycznych wprowadzono jedno odstępstwo od tej reguły. Dotyczy ono nawiasu okrągłego otwierającego i zamykającego, który jest separatorem, a w dwóch przypadkach jest również samodzielnym słowem. Pierwszy przypadek dotyczy nawiasu umieszczonego na liście, a drugi nawiasu użytego poza wyrażeniem.
Przykłady:

```
?pisz długość [()]
2
?(
Nie wiem jak zrobić (.
```

m oznacza ciąg cyfr reprezentujący część całkowitą liczby; jeśli nie występuje, domyślnie jest przyjmowane 0;

. oznacza kropkę dziesiętną (odpowiednik przecinka dziesiętnego);

n oznacza ciąg cyfr reprezentujący część ułamkową liczby; jeśli nie występuje, domyślnie jest przyjmowane 0;

E lub e jest znakiem oznaczającym początek zapisu wykładnika, można go zatem interpretować jako "razy dziesięć do potęgi";

v oznacza znak wykładnika + lub −; jeśli nie występuje, to znakiem wykładnika jest + ;

w oznacza ciąg cyfr reprezentujących wykładnik liczby, jeśli nie występuje, domyślnie jest przyjmowane 0.

Wymaga się, aby liczba w postaci dziesiętnej zawierała przynajmniej część *m* lub *.n* ; ponadto jeśli wystąpi kropka dziesiętna, to musi wystąpić część *n*, a jeśli wystąpi znak E, to musi wystąpić także część *w*.

Przykłady

Zapis liczby	Wartość liczby
+7	7
−0999	−999
0107	107
000	0
+0	0
−0	0
.0	0
.1	0.1
−.01	−0.01
−.01E2	−1
1.3E−2	0.013

Liczbami szesnastkowymi są ciągi znaków w postaci:

s#*h*

gdzie:

s oznacza znak liczby + lub −; jeśli nie występuje, to znakiem liczby jest +;

h jest ciągiem cyfr dziesiętnych oraz małych lub wielkich liter : a,b,c,d,e,f. Litery te reprezentują tu kolejne, następujące po 9 cyfry szesnastkowe.

Przykłady

Zapis liczby szesnastkowej	Wartość liczby
−#ff	−255
#a1	161
+#12	18

Należy podkreślić, że nie ma różnicy w reprezentacji wewnętrznej liczb całkowitych i ułamkowych oraz dziesiętnych i szesnastkowych. Wszystkie liczby są pamiętane w znormalizowanej postaci zmiennopozycyjnej. Oznacza to, że dokładność przedstawiania liczb wynosi około 15 cyfr znaczących (dokładność ta nie wpływa na szybkość obliczeń, można ją zmieniać w zakresie od 1 do 16 za pomocą poleceń pliku konfiguracyjnego). Zakres reprezentacji liczb dodatnich wynosi od 1.7E−308 do 1.7E+308, a ujemnych od −1.7E+308 do 1.7E−308.

W AC-LOGO wartości logiczne są reprezentowane przez dwa słowa: PRAWDA (ang. TRUE) i FAŁSZ (ang. FALSE). Słowa te nie są zastrzeżone i w zależności od kontekstu mogą być interpretowane jako wartości logiczne lub tak jak inne słowa. Jeżeli mają one reprezentować wartości logiczne, to zasadniczo powinny być pisane wielkimi literami alfabetu. Jeżeli nie są, to w kontekście, w którym jest wymagana wartość logiczna, AC-LOGO przeprowadza automatyczną konwersję do wielkich liter.

Zauważmy jeszcze, że w LOGO istnieją funkcje pierwotne o nazwach: prawda i fałsz, których wartościami są odpowiednio PRAWDA i FAŁSZ (por. rozdział 8).

Słowo poprzedzone cudzysłowem — operatorem dosłowności (por. punkt 3.5) — jest traktowane jako tekst. W szczególności, separator poprzedzony cudzysłowem jest tekstem pustym — słowem pustym. W AC-LOGO liczba także może być poprzedzona cudzysłowem, co nie zmienia jej znaczenia.

Słowo poprzedzone dwukropkiem oznacza — w zależności od kontekstu — wartość zmiennej, której nazwą jest to słowo (por. p. 3.9) lub parametr formalny procedury (por. p. 3.6). W nazwach zmiennych i parametrów formalnych procedur wielkie i małe litery alfabetu nie są rozróżniane.

W AC-LOGO słowo będące elementem listy jest traktowane jako tekst, niezależnie od tego czy jest poprzedzone operatorem dosłowności — cudzysłowem, czy nie. Wyjątkiem jest słowo będące elementem listy stanowiącej parametr aktualny procedury pierwotnej wymagającej listy wywołań procedur jako parametru (por. rozdział 8 — opis procedur JEŚLI, POWTÓRZ i ZRÓB — ang. IF, REPEAT i RUN).

W każdym innym kontekście słowo jest traktowane jako wywołanie procedury (por. punkt 3.6). W nazwach procedur wielkie i małe litery alfabetu nie są rozróżniane.

Przykłady

"PRAWDA	w zależności od kontekstu oznacza wartość logiczną PRAWDA lub tekst PRAWDA;
prawda	oznacza wywołanie funkcji pierwotnej PRAWDA;
"Ala!"#	oznacza tekst Ala!"# ;
"""	oznacza tekst " ;
""	oznacza słowo puste;
'Ala i Ola'	oznacza wywołanie procedury o nazwie Ala i Ola;
"'Ala i Ola'	oznacza tekst Ala i Ola;
:Ala!"#	oznacza wartość zmiennej o nazwie ALA!"# ;
[Ala!"#]	oznacza listę zawierającą słowo Ala!"# ;
["Ala!"#]	oznacza listę zawierającą słowo Ala!"# ;
Ala!"#	oznacza wywołanie procedury ALA!"# .

Listy

Lista jest ciągiem słów lub list ujętym w nawiasy kwadratowe []. W szczególności, ciąg ten może mieć długość równą zeru. Mówimy wówczas o liście pustej. Listy stanowiące elementy innych list nazywamy podlistami.

Przykłady

[Ala Ola]	dwuelementowa lista, której elementami są słowa: Ala i Ola;
[]	lista pusta;
["]	jednoelementowa lista, której elementem jest słowo puste;
[""]	jednoelementowa lista, której elementem jest słowo puste;
["""]	jednoelementowa lista, której elementem jest słowo puste;
[[][[]]]	dwuelementowa lista, której elementami są: lista pusta oraz lista zawierająca listę pustą;
[111 25.3E2 []]	trójelementowa lista, której elementami są: liczby 111 i 25.3E3 oraz lista pusta;
[(3 + 4)]	pięcioelementowa lista, której elementami są: (, 3, +, 4 i).

3.5. Wyrażenia

Wyrażenie jest formułą do obliczania wartości. W ogólności, wyrażenie składa się z argumentów i operatorów. W skrajnym przypadku wyrażenie

może składać się z jednego argumentu. Argumentami wyrażenia mogą być: słowa, w szczególności liczby i wartości logiczne, listy, zmienne, wywołania funkcji lub wyrażenia ujęte w nawiasy okrągłe. Kolejność wykonywania operacji w wyrażeniu jest określona przez priorytety i reguły łączności występujących w nim operatorów. W pierwszej kolejności są wykonywane operacje o najwyższym priorytecie. W przypadku równych priorytetów kolejnych operatorów operacje są wykonywane od strony lewej do prawej, gdy operatory są lewostronnie łączne, a od prawej do lewej, gdy operatory są łączne prawostronnie. Kolejność wykonywania operacji można ustalać jawnie porządkując wyrażenie za pomocą nawiasów okrągłych.

Operatory dzielą się na jedno- i dwuargumentowe. Niezależnie od powyższej klasyfikacji, wyróżniamy operatory specjalne, arytmetyczne i operatory relacji.

Do operatorów specjalnych zaliczamy:
— operator dosłowności "
— operator wartościowania :
Operatorów tych nie oddziela się separatorem od argumentów. Operatory specjalne nie mogą być składane ze sobą.

Argumentem operatora dosłowności jest słowo. Operator dosłowności zapobiega interpretacji tego słowa jako wywołania procedury.

Argumentem operatora wartościowania jest słowo będące nazwą zmiennej. Operator ten powoduje pobranie wartości tej zmiennej.

Przykłady

Wyrażenie	Wartość wyrażenia
"Ala	słowo trójznakowe Ala
"\Ala	słowo trójznakowe Ala
"'Ala'	słowo trójznakowe Ala
""'Ala	słowo czteroznakowe "Ala
":Ala	słowo czteroznakowe :Ala
:Ala	wartość zmiennej o nazwie ALA
:"Ala	wartość zmiennej o nazwie "ALA
::Ala	wartość zmiennej o nazwie :ALA

W Logo są dostępne następujące operatory arytmetyczne: jeden operator jednoargumentowy:
minus —
oraz cztery operatory dwuargumentowe:
dodawania +
odejmowania —

mnożenia	*
dzielenia	/

Operacje arytmetyczne można wykonywać tylko na argumentach o wartościach liczbowych. Jednoargumentowy operator minus jest odpowiednikiem znaku liczby dla wartości zmiennych i funkcji. Operatora tego nie oddziela się separatorem od argumentu i nie można go składać z samym sobą.

Przykłady

Wyrażenie	Wartość wyrażenia
1	1
sin 0	0
1 + 2 * 5	11
(1 + 2) * 5	15
#a / 2	5
1 − −1	2
1 − (−1)	2
(((1 + 2) * 5 − 4) * 2 + 1) * 4	92
2 * pi	6.28318530717959
1 + (sin 90) * (cos 90)	1
(sin 45) * (sin 45)	0.5
sin (45 * sin 45)	0.527249542282299
sin 45 * sin 45	0.527249542282299
−x	zanegowana wartość funkcji x
−:x	zanegowana wartość zmiennej x

W AC-LOGO są dostępne następujące operatory relacji:

operator równości	=
operator nierówności	<>
operator większości	>
operator mniejszości	<
operator większości lub równości	>= lub =>
operator mniejszości lub równości	<= lub =<

Wynikiem operacji relacji jest wartość logiczna.

Operatory równości oraz nierówności mogą działać na dowolnych argumentach. Argumenty są równe, jeżeli są tego samego typu i ich kolejne elementy są równe.

Pozostałe operacje relacji można wykonywać na argumentach, których wartości są słowami. Porównywanie słów przebiega w następujący sposób. Najpierw sprawdza się, czy oba porównywane słowa są liczbami i jeśli tak, to porównuje się ich wartości. Natomiast jeśli nie, to porównuje się słowa

w porządku leksykograficznym, czyli porównuje się kody ASCII kolejnych znaków obu słów. Porównanie kończy się w momencie wykrycia różnych znaków lub końca jednego ze słów. W trakcie porównywania, wartości kodów ASCII znaków narodowych są zmieniane tak, aby odpowiadały porządkowi leksykograficznemu (tzn. A < Ą < B ...).

Przykłady

Wyrażenie	Wartość wyrażenia
1 = "1	PRAWDA
1 = 2	FAŁSZ
(1 + 1) = 2	PRAWDA
7 = "Ala	FAŁSZ
8 = (8)	PRAWDA
8 = [8]	FAŁSZ
8 = [Ala]	FAŁSZ
"Ala <> [Ala]	PRAWDA
"Ala = pierw [Ala]	PRAWDA
[Ala] = [[Ala]]	FAŁSZ
[Ala] <> [Ala[]]	PRAWDA
[Ala Ola] = [Ola Ala]	FAŁSZ
(2 + 5) >= (1 + 4)	PRAWDA
"Sto > 100	PRAWDA
"Alan > "Ala	PRAWDA
"Alan > "Ola	FAŁSZ
"Bąk > "Bak	PRAWDA
"Bąk > "Bbk	FAŁSZ
"Ala > "ala	FAŁSZ
równe? pierw [2 1] ost [2 2]	PRAWDA
pierw [2 1] = ost [2 2]	F
(prawda = 2) > 1	PRAWDA
1 = 1 = 1	FAŁSZ

Dokładniejszego omówienia wymagają trzy ostatnie przykłady. W pierwszym z nich parametrem aktualnym funkcji pierw jest wyrażenie: [2 1] = ost [2 2]. Wartością tego wyrażenia jest wartość logiczna FAŁSZ, gdyż lista [2 1] nie jest równa liczbie 2. Wywołanie funkcji PIERW z parametrem FAŁSZ daje w wyniku pierwszy element tego słowa czyli F.

W przedostatnim przykładzie wywołanie funkcji prawda daje w wyniku słowo PRAWDA, które nie jest równe słowu 2. Wartością wyrażenia w nawiasie jest więc słowo FAŁSZ. Kod ASCII pierwszego znaku tego słowa ma wartość #46, jest zatem większy od kodu ASCII pierwszego znaku

słowa 1, który wynosi #31. Otrzymujemy zatem ostateczną wartość wyrażenia — PRAWDA.

W ostatnim przykładzie wynikiem porównania 1 = 1 jest wartość logiczna PRAWDA, która nie jest równa liczbie 1. Wartością wyrażenia jest zatem FAŁSZ.

Wszystkie dostępne w AC-LOGO operatory zebrano w tabeli 3.1. Priorytety operatorów zmieniają się od 1 do 6, gdzie 1 jest najwyższym priorytetem, a 6 — najniższym. Typy argumentów i wyniku są zaznaczone w następujący sposób:

s — dowolne słowo;
n — liczba;
b — wartość logiczna;
l — lista.

W przypadku, gdy argumenty lub wynik mogą przyjmować wartości dwóch typów, są wymieniane oba dozwolone typy.
Kolejność wykonywania operacji w AC-LOGO zilustrowano poniżej.

Wyrażenie	Kolejność wykonywania
1 + 2 + −3	(1 + 2) + (−3)
1 + 2 * 3	1 + (2 * 3)
1 + 2 / 3 * 4	1 + ((2 / 3) * 4)
1 + 2 / 3 / 4	1 + ((2 / 3) / 4)
1 + 3 = 2 + 2	(1 + 3) = (2 + 2)
1 + sin 45 + 45	1 + (sin (45 + 45))
abs :x + sin :x	abs ((:x) + (sin (:x)))

Tablica 3.1. Priorytety i łączność operatorów

Priorytet	Operator	Nazwa	Typ argum.	Typ wyniku	Łączność
1	"	operator dosłowności	s	s	prawostronna
	:	operator wartościowania	s	sl	prawostronna
2	−	jednoargumentowy minus	n	n	prawostronna
3	*	mnożenie	n	n	lewostronna
	/	dzielenie	n	n	lewostronna
4	+	dodawanie	n	n	lewostronna
	−	odejmowanie	n	n	lewostronna
5	=	równość	sl	b	lewostronna
	>	większość	s	b	lewostronna
	<	mniejszość	s	b	lewostronna
	>= lub =>	większość lub równość	s	b	lewostronna
	<= lub =<	mniejszość lub równość	s	b	lewostronna
6		wywołanie funkcji	s	sl	prawostronna

Na zakończenie podamy przykłady wyrażeń błędnie skonstruowanych:

`--1`	Operacja jednoargumentowy minus nie może być złożona sama ze sobą.
`-(:x)`	Operator jednoargumentowy minus nie może być oddzielony separatorem od argumentu.
`1-1=0`	W wyrażeniu nie występują separatory, czyli jest ono jednym słowem, interpretowanym jako nazwa procedury — tu funkcji. W kontekście, w którym jest wymagana liczba (wartość logiczna), wystąpienie tego słowa, jeśli nie jest ono nazwą funkcji arytmetycznej (logicznej), jest błędem.
`1 -1 = 0`	Minus nie oddzielony spacją od liczby jest jej znakiem. Między 1 a −1 nie ma zatem operatora, czyli zapis ten przedstawia dwa wyrażenia — wartością pierwszego jest 1, a drugiego, tzn. −1 = 0 jest FAŁSZ.
`1 + ( 2`	Brak nawiasu zamykającego.
`"A + "la`	Argumentami operatora + mogą być tylko liczby.
`1 + * 2`	Kolejne operatory dwuargumentowe muszą być oddzielone argumentami.
`[Ala] > [Ola]`	Argumentem operatora większości nie może być lista.
`[Ala] >= "Ola`	Jak wyżej.
`[Ala] < 1`	Jak wyżej.
`pierw [2 1] > ost [2 2]`	Parametrem aktualnym funkcji PIERW jest wyrażenie [2 1] > ost [2 2]. Wyrażenie to jest błędne, gdyż użycie operatora większości do porównywania listy z liczbą jest niedozwolone.

3.6. Procedury

Procedura jest zapisem algorytmu w języku LOGO. Wyróżniamy deklarację procedury i wywołanie procedury. Procedura musi być zadeklarowana przed jej pierwszym wywołaniem.

Deklarację procedury rozpoczyna wywołanie procedury pierwotnej OTO (ang. TO), po którym następuje nazwa deklarowanej procedury oraz jej kolejne parametry formalne. Dalej następuje treść procedury zakończona wywołaniem procedury pierwotnej JUŻ (ang. END).

Nazwą procedury jest niepuste słowo. W nagłówku procedury słowo to

może, ale nie musi, być poprzedzone operatorem dosłowności (znakiem cudzysłowu). Nie zaleca się stosowania nazw procedur rozpoczynających się od cudzysłowu, dwukropka, znaku #, operatora ani cyfry. W danym programie nazwy procedur muszą być jednoznaczne.

Parametrem formalnym procedury może być dowolne niepuste słowo poprzedzone dwukropkiem (podobnie jak w przypadku nazw procedur nie zaleca się stosowania nazw parametrów formalnych rozpoczynających się od cudzysłowu, dwukropka, znaku #, operatora ani cyfry). Można deklarować procedury bezparametrowe. Treść procedury następuje wówczas po jej nazwie. Jeżeli procedura ma kilka parametrów formalnych, to ich nazwy muszą być różne.

Treść procedury rozpoczyna się od pierwszego nie poprzedzonego dwukropkiem słowa występującego za nazwą procedury. Treścią procedury jest ciąg wywołań procedur. W wywołaniach tych mogą występować parametry formalne deklarowanej procedury. W treści procedury może również wystąpić wywołanie jej samej. Mamy wówczas do czynienia z rekurencją.

Alternatywną metodą deklarowania procedury jest użycie procedury pierwotnej OKREŚL (ang. DEFINE; por. rozdział 8). Jej zaletami są: oszczędność pamięci oraz symetryczne traktowanie danych i procedur, co umożliwia pisanie programów generujących inne programy. W AC-LOGO ta metoda nie zapewnia jednak dostępu za pomocą edytorów do tekstu źródłowego tak zadeklarowanej procedury.

Przykłady. Deklarowanie procedur

```
?oto rozmówka
> pisz [Jak ma na imię dziewczyna, z którą chodzisz?]
> przypisz "imię czytajlistę
> jeśli puste? :imię
>   [stop]
> jeśli równe? "JESTEM wl pierw :imię
> [
>   pisz [Naprawdę? To smutne.]
>   stop
> ]
> pisz zdanie :imię [jest ładna.]
>już
ROZMÓWKA zdefiniowano

?oto tablica_pierwiastków :ile
> przypisz "n 1
> powtórz :ile
```

```
> [
>    (pisz :n "\ pierwiastek :n)
>    przypisz "n :n + 1
> ]
>już
TABLICA_PIERWIASTKÓW zdefiniowano
```

Wywołanie procedury polega na podaniu jej nazwy z następującymi za nią kolejnymi parametrami aktualnymi. Parametry aktualne są wyrażeniami. Wartości tych wyrażeń są przypisane parametrom formalnym na początku wykonania procedury. Liczba parametrów aktualnych musi być równa liczbie parametrów formalnych w deklaracji procedury.

Przykłady. Wywoływanie procedur

```
?rozmówka
Jak ma na imię dziewczyna, z którą chodzisz?
 Magda
Magda jest ładna.
?rozmówka
Jak ma na imię dziewczyna, z którą chodzisz?
 Jestem wrogiem kobiet.
Naprawdę? To smutne.

?tablica_pierwiastków 5
1 1
2 1.41421356237309
3 1.73205080756888
4 2
5 2.23606797749979
```

Wykonywanie procedur rekurencyjnych jest w pewnych przypadkach optymalizowane przez kompilator AC-LOGO. Dotyczy to tak zwanej *rekurencji ogonowej* (końcowej), z którą mamy do czynienia wówczas, gdy wywołanie rekurencyjne jest ostatnim wywołaniem w treści procedury przed wywołaniem procedury pierwotnej JUŻ. Optymalizacja polega na zastąpieniu rekurencji iteracjami. Nie są wówczas tworzone kolejne poziomy wywołań rekurencyjnych procedury i nie są rezerwowane kolejne obszary pamięci, jak w pozostałych przypadkach rekurencji. Zwiększa to szybkość działania procedur i umożliwia pisanie programów z nieskończoną rekurencją.

Poniżej przedstawiamy przykłady trzech procedur. W procedurze TROtest zastosowano rekurencję ogonową, a w procedurze nieTROtest — rekurencję nieogonową. Frazę:

```
odśmieć (pisz "poziom: :n "\ "wolne: wolne)
```

umieszczono w obu procedurach w celu ukazania liczby bajtów wolnej pamięci na każdym poziomie wywołania rekurencyjnego. Procedura pętla jest przykładem zastosowania nieskończonej rekurencji.

Przykłady

```
?oto TROtest :n
>jeśli :n > 5 [stop]
>odśmieć (pisz "poziom: :n "\ "wolne: wolne)
>TROtest :n + 1
>już
TROTEST zdefiniowano
?oto nieTROtest :n
>jeśli :n > 5 [stop]
>odśmieć (pisz "poziom: :n "\ "wolne: wolne)
>nieTROtest :n + 1
>wpisz "
>już
NIETROTEST zdefiniowano
?odśmieć TROtest 1
poziom:1 wolne:59888
poziom:2 wolne:59877
poziom:3 wolne:59877
poziom:4 wolne:59877
poziom:5 wolne:59877
?odśmieć nieTROtest 1
poziom:1 wolne:59885
poziom:2 wolne:59874
poziom:3 wolne:59863
poziom:4 wolne:59852
poziom:5 wolne:59841

?oto pętla
> jeśli klawisz? [przerwij]
> demonstracja
> pętla
>już
PĘTLA zdefiniowano
?pętla
{Wykonywanie procedury demonstracja wielokrotnie, aż do naciśnięcia
dowolnego klawisza}
```

3.7. Funkcje

Funkcja jest procedurą, która przekazuje wartość wyrażeniu, z którego ją wywołano. W deklaracji funkcji musi wystąpić co najmniej jedno wywołanie procedury pierwotnej WYNIK (ang. OUTPUT; por. rozdział 8). W trakcie wykonywania funkcji procedura pierwotna WYNIK musi być wykonana, aby funkcja otrzymała wartość. Nazwa funkcji nie może być liczbą.

Przykłady

```
?oto g
>wynik 9.81
>już
G zdefiniowano
?pisz g
9.81
?oto potęga :podstawa :wykładnik
>{wykładnik dodatni i całkowity}
> jeśli :wykładnik = 0
>    [wynik 1]
> inaczej
>    [wynik :podstawa * potęga :podstawa :wykładnik – 1 ]
>już
POTĘGA zdefiniowano
?pisz potęga 2 4
16
```

3.8. Procedury i funkcje pierwotne

Procedurami i funkcjami pierwotnymi nazywamy procedury i funkcje, które zostały wstępnie zadeklarowane w języku LOGO. Procedurę i funkcję pierwotną można zatem wywołać bez uprzedniej jawnej deklaracji. Procedur i funkcji pierwotnych nie można redefiniować.

Niektóre procedury i funkcje pierwotne mogą być wywoływane z dowolną liczbą parametrów aktualnych. Mówimy wówczas o wywołaniu zachłannym, a procedury (funkcje), które mogą być wywoływane zachłannie nazywamy procedurami (funkcjami) zachłannymi. W przypadku wywołania zachłannego nazwa procedury (funkcji) wraz z parametrami aktualnymi musi być ujęta w nawiasy okrągłe. W jednym wyrażeniu mogą występować

zarówno nawiasy okrągłe ustalające kolejność działań, jak i nawiasy okrągłe zaznaczające zachłanność.

Przykłady

```
?(pisz "a 1 2 1 + 2)
a123
?przypisz "i "b
?pisz (lub (:i = "a) (:i = "b) (:i = "c))
PRAWDA
```

3.9. Zmienne

Zmienna jest elementem języka posiadającym nazwę i wartość. Nazwą zmiennej może być dowolne niepuste słowo. Wielkie i małe litery w nazwach zmiennych nie są rozróżniane. Wartość zmiennej można zmieniać w trakcie wykonywania programu. Wartością zmiennej może być słowo lub lista. W szczególności, wartością zmiennej może być zatem liczba, wartość logiczna, nazwa zmiennej lub procedury. W LOGO dowolnej zmiennej można przypisać wartość dowolnego typu.

Utworzenia nowej zmiennej lub przypisania istniejącej zmiennej nowej wartości dokonuje się przez wykonanie procedury pierwotnej PRZYPISZ (ang. MAKE; por. rozdział 8).

Wartość zmiennej uzyskuje się albo przez podanie jej nazwy poprzedzonej operatorem wartościowania — dwukropkiem, albo przez wykonanie procedury pierwotnej WARTOŚĆ (ang. THING; por. rozdział 8).

Zmienne dzielą się na lokalne i globalne. Zmiennymi lokalnymi są parametry formalne procedur oraz zmienne zadeklarowane za pomocą procedury pierwotnej LOKALNA (ang. LOCAL; por. rozdział 8). Pozostałe zmienne są zmiennymi globalnymi. Każda zmienna jest scharakteryzowana przez zasięg, czyli fragment programu, w którym jest dostępna jej wartość. W AC-LOGO zasięgiem zmiennej lokalnej jest procedura, w której ją zadeklarowano, oraz te procedury z niej wywołane, w których nie zadeklarowano zmiennej lokalnej o tej samej nazwie. Podobnie, zasięgiem zmiennej globalnej jest cały program, z wyjątkiem tych procedur, w których zadeklarowano zmienną lokalną o tej samej nazwie. Zatem, jeżeli w procedurze wywołanej jest zadeklarowana zmienna lokalna o tej samej nazwie co zmienna w procedurze wywołującej, to "przesłania" ona zmienną z procedury wywołującej. Innymi słowy, w procedurze wywołanej jest dostępna wartość zmiennej lokalnej w niej zadeklarowanej.

Przykłady

```
?oto a :i
> pisz zdanie [W procedurze A zmienna I ma wartość] :i
> b :i + 1
> c :i + 2
>już
A zdefiniowano
?oto b :i
> pisz zdanie [W procedurze B zmienna I ma wartość] :i
>już
B zdefiniowano
?oto c :j
> pisz zdanie [W procedurze C zmienna I ma wartość] :i
> pisz zdanie [W procedurze C zmienna J ma wartość] :j
>już
C zdefiniowano
?c 0
I nie ma zdefiniowanej wartości w procedurze C.
?przypisz "i 1
?a 2
W procedurze A zmienna I ma wartość 2
W procedurze B zmienna I ma wartość 3
W procedurze C zmienna I ma wartość 2
W procedurze C zmienna J ma wartość 4
?b 2
W procedurze B zmienna I ma wartość 2
?c 2
W procedurze C zmienna I ma wartość 1
W procedurze C zmienna J ma wartość 2
```

3.10. Przekazywanie parametrów do procedur

W LOGO parametry są przekazywane do procedur przez wartość. W trakcie wywołania procedury z parametrami następuje obliczenie wartości wyrażeń stanowiących jej parametry aktualne i przypisanie ich parametrom formalnym. Parametry formalne można zatem traktować jak zmienne lokalne o wartościach początkowych zainicjowanych za pomocą parametrów aktualnych. Wartością parametru aktualnego procedury może być słowo

będące nazwą zmiennej globalnej lub zmiennej lokalnej zadeklarowanej w procedurze wywołującej. W treści procedury wywołanej są możliwe wówczas zarówno operacje na tej nazwie, jak i na wartości zmiennej przez nią reprezentowanej. Służą do tego procedury pierwotne WARTOŚĆ i PRZYPISZ (ang. THING i MAKE; por. rozdział 8). W szczególności, uzyskujemy dzięki temu możliwość przekazywania wyników procedury wywoływanej do procedury wywołującej za pomocą parametrów. Warto bowiem zauważyć, że jeśli parametrem aktualnym jest nazwa zmiennej, to w procedurze wywołanej nie jest tworzona kopia wartości tej zmiennej. Zatem, jeśli parametrem aktualnym przekazanym do procedury jest nazwa zmiennej, to pomimo, że formalnie przekazanie parametru odbywa się przez wartość, w istocie jest ono przekazaniem przez zmienną.

Wartością parametru aktualnego procedury może być również słowo będące nazwą procedury. Możliwe są wówczas operacje na tej nazwie, operacje na treści procedury przez nią reprezentowanej oraz jej wykonywanie. Do wykonywania operacji na treści procedury służą procedury pierwotne TREŚĆ (ang. TEXT) i OKREŚL (ang. DEFINE). Do wykonywania procedury służy procedura pierwotna ZRÓB (ang. RUN). Zatem, jeśli przekazanym do procedury parametrem jest nazwa procedury, która będzie następnie wykonywana, to pomimo, że formalnie przekazanie parametru odbywa się przez wartość, w istocie jest ono przekazaniem przez procedurę.

Poniżej przedstawiamy trzy przykłady. Pierwszy przykład, który stanowią dwie procedury: przekazanie_przez_wartość i przekazanie_przez_zmienną ilustruje różnice między przekazaniem parametru przez wartość, a przekazaniem przez zmienną. Obydwie procedury w nieco zawoalowany sposób powodują wyświetlenie listy będącej ich parametrem bez ostatniego elementu. Efektem ubocznym drugiej procedury jest zmiana wartości zmiennej globalnej LISTA. Frazę:

```
odśmieć (pisz "wolne: wolne)
```

umieszczono w obu procedurach dwukrotnie w celu wyświetlania liczby bajtów wolnej pamięci bezpośrednio po przekazaniu parametrów i po wykonaniu treści procedury.

W drugim przykładzie, który stanowią dwie procedury: ini_pseudo_tab i wyświetl_pseudo_tab; jest on ilustracją zastosowania metody przekazywania parametru przez zmienną do konstruowania struktur danych symulujących tablice.

Trzeci przykład — procedura tabela — ukazuje siłę wyrazu metody przekazania parametru przez procedurę.

Przykłady

```
?oto przekazanie_przez_wartość :lista
> odśmieć (pisz "wolne: wolne)
> przypisz "lista bezost :lista
> pisz :lista
> odśmieć (pisz "wolne: wolne)
>już
PRZEKAZANIE_PRZEZ_WARTOŚĆ zdefiniowano
?oto przekazanie_przez_zmienną :nazwa
>{Parametr aktualny powinien być nazwą zmiennej}
> odśmieć (pisz "wolne: wolne)
> przypisz :nazwa bezost wartość :nazwa
> pisz wartość :nazwa
> odśmieć (pisz "wolne: wolne)
>już
PRZEKAZANIE_PRZEZ_ZMIENNĄ zdefiniowano
?przypisz "lista [To jest krótka lista pięcioelementowa]
?przekazanie_przez_wartość :lista
wolne:59980
To jest krótka lista
wolne:59967
?pisz :lista
To jest krótka lista pięcioelementowa
?przekazanie_przez_zmienną "lista
wolne:59980
To jest krótka lista
wolne:59994
?pisz :lista
To jest krótka lista
?oto ini_pseudo_tab :nazwa :rozmiar
>{parametr :nazwa identyfikuje pseudo tablicę}
>{przekazanie odpowiada przekazaniu "przez zmienną"}
>{:rozmiar > 0}
> powtórz :rozmiar
> [
>    przypisz słowo :nazwa :rozmiar 0
>    przypisz "rozmiar :rozmiar — 1
> ]
>już
```

```
INI_PSEUDO_TAB zdefiniowano
?oto wyświetl_pseudo_tab :nazwa :rozmiar
>{parametr :nazwa identyfikuje pseudo tablicę}
>{przekazanie odpowiada przekazaniu "przez zmienną"}
>{:rozmiar > 0}
> powtórz :rozmiar
> [
>   wpisz słowo :nazwa :rozmiar
>   (pisz "\ =\ wartość słowo :nazwa :rozmiar)
>   przypisz "rozmiar :rozmiar – 1
> ]
>już
WYŚWIETL_PSEUDO_TAB zdefiniowano
?ini_pseudo_tab "tablica 5
?pokaż :tablica
Zmienna TABLICA nie ma zdefiniowanej wartości.
?wyświetl_pseudo_tab "tablica 5
tablica5 = 0
tablica4 = 0
tablica3 = 0
tablica2 = 0
tablica1 = 0
?oto tabela :funkcja
>{parametr :funkcja jest nazwą funkcji}
>{przekazanie tego parametru odpowiada przekazaniu
> "przez procedurę"}
> lokalna "x
> przypisz "x 0
>
> jeśli i nie pierwotne? :funkcja nie określone? :funkcja
> [
>   (pisz [Nie umiem zrobić tabeli] :funkcja)
>   stop
> ]
> (pisz "" x  ' :funkcja "\ \(x\))
> wpisz "" '
> powtórz 10
> [
>   (pisz :x ""  ' zrób zd :funkcja :x)
>   przypisz "x :x + 10
```

```
> ]
>już
TABELA zdefiniowano
?tabela "cos
 x   cos (x)
 0   1
10   0.984807753012208
20   0.939692620785908
30   0.866025403784439
40   0.766044443118978
50   0.642787609686539
60   0.5
70   0.342020143325669
80   0.17364817766693
90   0
?tabela "superhipersinus
Nie umiem zrobić tabeli superhipersinus
```

4. SYSTEM PROGRAMOWANIA AC-LOGO

AC-LOGO stanowi system programowania, w którym interakcyjny kompilator jest ściśle powiązany z edytorami i systemem okienkowym. Dzięki temu po wywołaniu AC-LOGO można przeprowadzać edycję, kompilację, uruchamianie oraz wykonywanie programów. AC-LOGO zostało skonstruowane w dwóch wersjach językowych — polskiej i angielskiej. Obie wersje są identyczne pod względem funkcjonalnym, różnią się jedynie nazwami procedur pierwotnych.

Na system programowania AC-LOGO składają się następujące pliki: ACLOGO.EXE (system LOGO w wersji polskiej), ACLOGOA.EXE (wersja angielska), ACLOGO.CFG, LOGO.THP lub LOGOA.THP, LOGO.GHP, SYSTEM.FNT i LAT8X14.FNT. Pliki te zawierają odpowiednio: kompilator w wersji polskiej i angielskiej, plik konfiguracyjny, pliki z tekstami i rysunkami podpowiedzi (pomocy) oraz pliki z dwoma podstawowymi generatorami znaków. Dodatkowo, dyskietka dystrybucyjna zawiera plik tekstowy LOGO.DOC zawierający informacje o zmianach wprowadzonych w kolejnych wersjach systemu programowania AC-LOGO, począwszy od wersji 1.1.

W dalszej części rozdziału opisujemy wyłącznie polską wersję systemu. Opis użytkowania wersji angielskiej uzyskuje się z poniższego opisu przez zamianę odpowiednich polskich nazw procedur i plików na nazwy angielskie.

Parametry wywołania systemu AC-LOGO są pamiętane w pliku konfiguracyjnym o nazwie ACLOGO.CFG. Plik ten, jeśli ma być odczytany przez system, powinien znajdować się w katalogu aktualnym (to znaczy w tym katalogu, z którego wywołujemy system LOGO). Dokładny opis tego pliku zawarto w Dodatku F. Przy pomocy poleceń zawartych w tym pliku można zmieniać niektóre parametry systemu. Plik konfiguracyjny jest wczytywany jako pierwszy, bezpośrednio po uruchomieniu systemu AC-LOGO. Jeśli pliku konfiguracyjnego nie ma w katalogu aktualnym (lub jest on pusty), to są przyjmowane tak zwane parametry domyślne, które opisano w Dodatku F.

Sposób wywołania systemu programowania AC-LOGO zależy od trybu pracy mikrokomputera.

W celu wywołania AC-LOGO należy wpisać polecenie ACLOGO, ewentualnie z dwoma lub więcej parametrami oddzielonymi spacją. Parametry te mogą być jawne, domyślne lub mogą być pominięte. Parametrami są nazwy plików zawierających teksty źródłowe procedur LOGO. Ostatnim parametrem z listy parametrów powinna być nazwa jednej z procedur zawartych w którymkolwiek z wczytanych wcześniej plików. Po wywołaniu systemu programowania AC-LOGO z dwoma lub więcej parametrami następuje załadowanie do pamięci operacyjnej treści procedur znajdujących się w plikach, których nazwy stanowią parametry wywołania, ich kompilacja i wykonanie programu począwszy od procedury, której nazwa jest ostatnim parametrem wywołania. Należy zauważyć, że wczytywane pliki powinny znajdować się w katalogu zawierającym inne pliki źródłowe LOGO — nazwę tego katalogu można zmienić przy pomocy poleceń pliku konfiguracyjnego.

Jeśli na dysku w katalogu aktualnym znajduje się plik o nazwie AUTOEXEC.LOG, to procedury z tego pliku są wczytywane automatycznie po wczytaniu plików — parametrów wywołania systemu. Plik AUTOEXEC.LOG powinien zawierać program w LOGO, którego wykonanie rozpoczyna się od procedury o nazwie AUTOEXEC. Nazwę pliku AUTOEXEC.LOG oraz nazwę procedury startowej AUTOEXEC można zmienić, podając inne nazwy w pliku konfiguracyjnym.

Jeśli w katalogu aktualnym, z którego wywołano program ACLOGO nie ma pliku AUTOEXEC.LOG, a wywołanie systemu programowania LOGO jest bezparametrowe, to na ekranie zostaje wyświetlona czołówka programu, a następnie jego nagłówek i znak zgłoszenia edytora wierszowego. System jest przygotowany do wprowadzania kolejnych procedur LOGO.

Przedstawimy teraz działanie części składowych systemu AC-LOGO. Po zgłoszeniu się system ten umożliwia:

- wprowadzanie tekstu procedur za pomocą edytora wierszowego lub systemu okienkowego,
- wywołanie edytora ekranowego,
- ładowanie procedur z dysku i ich składowanie na dysku,
- przeglądanie katalogu,
- kompilację procedur,
- wykonywanie procedur, połączone ewentualnie z ich śledzeniem,
- drukowanie tekstów procedur i zawartości ekranu,
- zakończenie pracy i przejście do systemu operacyjnego MS-DOS.

Działanie edytora wierszowego i ekranowego przedstawiono w rozdziałach 5 i 6, natomiast sposób korzystania z systemu okienkowego — w rozdziale 7.

Ładowanie tekstu procedur z dysku umożliwia między innymi procedura pierwotna ŁADUJ (ang. LOAD). Podczas ładowania nowych procedur,

procedury uprzednio zadeklarowane nie są niszczone, z wyjątkiem procedur o tych samych nazwach co procedury ładowane. Składowanie na dysku wszystkich procedur znajdujących się w danej chwili w pamięci operacyjnej mikrokomputera umożliwia procedura pierwotna ZAPISZ (ang. SAVEALL). Parametrem obu powyższych procedur jest nazwa pliku dyskowego. Nazwa ta, podobnie jak w systemie operacyjnym MS-DOS, składa się z nazwy dysku, nazwy podstawowej oraz rozszerzenia (por. [11]). AC-LOGO zezwala na stosowanie domyślnej nazwy dysku i domyślnego rozszerzenia nazwy. Dla plików zawierających teksty procedur rozszerzeniem domyślnym jest LOG, dla plików zawierających wartości zmiennych — TXT, dla plików pamiętających zawartość pamięci ekranu — SCR, natomiast dla plików z binarną zawartością obszaru pamięci — COD. Rozszerzenia domyślne mogą być zmienione na inne przy pomocy poleceń z pliku konfiguracyjnego, opisanych w dodatku F.

AC-LOGO umożliwia przeglądanie katalogu dysku. Służą do tego: odpowiednie procedury systemu okienkowego (por. rozdział 7) i procedury pierwotne KATALOG i PLIKI (ang. CAT i FCAT). W tym drugim przypadku jest możliwe stosowanie jako parametru niejednoznacznej nazwy pliku z użyciem znaków "*" i "?" (por. [11]).

Kompilacja procedur w AC-LOGO następuje automatycznie w trzech przypadkach:

- po zakończeniu deklarowania procedury za pomocą edytora wierszowego;
- po zakończeniu edycji procedur za pomocą edytora ekranowego;
- po załadowaniu z dysku do pamięci operacyjnej tekstu źródłowego procedur.

AC-LOGO umożliwia także jawne wywołanie kompilatora. Służy do tego procedura pierwotna OKREŚL (ang. DEFINE).

W wyniku kompilacji procedury powstaje jej postać półskompilowana, umożliwiająca następnie wykonanie tej procedury. Kompilacja procedury nie powoduje usunięcia jej wersji źródłowej — w pamięci operacyjnej mikrokomputera procedura jest przechowywana równocześnie w obu postaciach. Należy podkreślić, że w AC-LOGO jednostką kompilacji jest procedura, a nie program.

Uprzednio skompilowane procedury można wywoływać. Wywołanie procedur polega na podaniu ich nazw wraz z parametrami aktualnymi oraz znaku nowego wiersza (CR, ENTER). Liczba parametrów aktualnych w wywołaniu procedury musi być równa liczbie parametrów formalnych w jej deklaracji.

Wykonywanie procedury może być zawieszone przez wciśnięcie klawisza PAUSE na klawiaturze (i wznowione po naciśnięciu dowolnego klawisza)

oraz przerwane po wciśnięciu kombinacji klawiszy CTRL i BREAK. Wykonywanie programu jest także zawieszane na czas wciśnięcia klawisza SCROLL LOCK. Przerwanie wykonywania programu powoduje wyświetlenie komunikatu: "*** Przerwanie ***" i powrót do edytora wierszowego.

W trakcie wykonywania procedur może dojść do sytuacji, w której zostanie zajęta cała dostępna pamięć operacyjna mikrokomputera. Wykonywanie procedur jest wtedy zawieszane, a sterowanie jest przekazywane do procedury odzyskującej nieużytki pamięci przez jej uporządkowanie. Po uporządkowaniu pamięci wykonywanie procedur jest kontynuowane. AC-LOGO umożliwia także jawne wywoływanie procedury odzyskującej nieużytki. Służy do tego procedura pierwotna ODŚMIEĆ (ang. RECYCLE). Porządkowanie pamięci jest sygnalizowane obracającym się znakiem kreski w lewym, górnym rogu ekranu. Zmianę sygnalizacji porządkowania umożliwia procedura pierwotna .ODŚMIECANIE (ang. .INSTALL) oraz odpowiednie polecenie z pliku konfiguracyjnego.

System LOGO można opuścić wykonując procedurę pierwotną DOŚĆ (ang. BYE) lub .SYSTEM. Procedura DOŚĆ upewnia się co do podjętej decyzji pytając:

"Jesteś pewien (t/n)?"

Odpowiedzią pozytywną jest naciśnięcie klawisza T. Następuje po niej przejście do systemu operacyjnego MS-DOS. Po każdej innej odpowiedzi następuje powrót do aktualnie wykonywanej procedury bez utraty tekstu procedur i zmiennych.

4.1. Programy tłumaczące

System programowania AC-LOGO został wyposażony w szereg programów, które służą do tłumaczenia plików źródłowych (zawierających procedury w języku LOGO) oraz plików z rysunkami. Do programów tych należą:

- JUN2LAT — program tłumaczący kody znaków standardu ELWRO 800 Junior na kody znaków LATIN-II; służy on przede wszystkim do tłumaczenia kodów polskich znaków narodowych znajdujących się w plikach LOGO w standardzie Juniora na kody zastosowane w systemie AC-LOGO,
- LAT2JUN — program tłumaczący kody znaków standardu LATIN-II na kody znaków ELWRO 800 Junior (ma odwrotne działanie do poprzedniego programu),

- POL2ANG — program tłumaczący procedury zapisane w polskojęzycznej wersji AC-LOGO na wersję angielskojęzyczną; program może także służyć do tłumaczenia plików źródłowych Junior-LOGO,
- ANG2POL — program tłumaczący procedury zapisane w angielskojęzycznej wersji AC-LOGO na wersję polskojęzyczną; podobnie jak poprzedni, program ten może także służyć do tłumaczenia plików źródłowych Junior-LOGO,
- JUN2EGA — program tłumaczący pliki zawierające rysunki w formacie Junior-LOGO na pliki w formacie karty graficznej EGA/VGA; rysunki te mogą być także wczytywane na komputerze wyposażonym w kartę graficzną Hercules HGC,
- EGA2HGC — program tłumaczący pliki zawierające rysunki w formacie karty graficznej EGA/VGA na format karty Hercules HGC; należy zauważyć, że istnieje różnica między wczytywaniem rysunku w formacie EGA/VGA i w formacie HGC na komputerze wyposażonym w kartę graficzną HGC; pliki zawierające rysunki w formacie karty graficznej EGA/VGA mogą być wczytywane i wyświetlane na karcie graficznej HGC — wszystkie kolory o numerach większych od zera są wtedy traktowane jako kolory "białe" — widoczne na ekranie, tylko kolor 0 jest kolorem "czarnym"; program tłumaczący umożliwia ustawienie indywidualnie dla każdego koloru, czy jest on "czarny" czy "biały" (por. opcja "PC" w opisie pliku konfiguracyjnego w Dodatku F),
- PXL2HGC — program tłumaczący pliki w formacie PXL (standard plików z rysunkami monochromatycznymi większości systemów operacyjnych, między innymi systemu UNIX — tak zwana "mapa bitowa") oraz pliki zawierające tak zwane "ikony" różnych systemów okienkowych (na przykład X11 Windows) systemu operacyjnego UNIX na pliki w formacie karty graficznej HGC,
- HGC2PXL — program tłumaczący pliki w formacie karty graficznej Hercules HGC nas format PXL; w celu przetłumaczenia pliku w formacie karty EGA/VGA na format PXL należy najpierw taki plik przetłumaczyć na format karty Hercules HGC,
- HGC2EGA — program tłumaczący pliki w formacie karty graficznej Hercules HGC na format karty EGA/VGA — punktom

"czarnym" i "białym" obrazu w formacie HGC można przyporządkować numery dwóch kolorów dostępnych w formacie EGA/VGA; w ten sposób rysunek czarno-biały można zmienić na rysunek na przykład zielono-czerwony; numery odpowiadające punktom "białym" i "czarnym" można podać jako trzeci i czwarty parametr wywołania programu — w przypadku braku któregoś z tych parametrów są przyjmowane odpowiednio kolory 15 i 0.

Wszystkie powyższe programy mogą być wywoływane na dwa sposoby. Pierwszy z nich polega na wywołaniu programu z jednym parametrem — nazwą pliku. Nazwa ta jest traktowana najpierw jako nazwa pliku źródłowego — poddawanego tłumaczeniu. Po przetłumaczeniu zawartości pliku wynik tłumaczenia jest zapisywany do nowo utworzonego pliku o takiej samej nazwie, jak plik źródłowy — poprzednia zawartość tego pliku jest bezpowrotnie tracona. Drugi sposób polega na wywołaniu programu z dwoma nazwami plików — pierwsza z nich jest wtedy traktowana jako nazwa pliku źródłowego, a druga — nazwa pliku wynikowego. W takim przypadku plik źródłowy pozostaje niezmieniony.

Program HGC2EGA może być wywołany z trzema lub czterema parametrami. Jeżeli program posiada trzy parametry, to ostatni parametr jest traktowany jako numer koloru, na który mają być zamienione punkty "białe". W przypadku podania czterech parametrów, dwa ostatnie parametry są traktowane jako numery kolorów odpowiadających punktom odpowiednio "białym" i "czarnym". Numery kolorów muszą być podane w postaci ciągu cyfr dziesiętnych — wartość parametru jest zawsze niejawnie obcinana do zakresu od 0 do 15. Na przykład, wywołanie

```
HGC2EGA obraz.hgc obraz.ega 10 2
```

oznacza przekodowanie rysunku w formacie karty HGC z pliku OBRAZ.HGC do pliku OBRAZ.EGA, przy czym punkty "białe" są zamieniane na punkty w kolorze 10, a punkty "czarne" — na punkty w kolorze 2.

Jeżeli w katalogu, z którego wywołano program tłumaczący, znajduje się plik LOGO.CFG, to jest on automatycznie wczytywany i traktowany jako plik konfiguracyjny programu (por. opis pliku konfiguracyjnego w Dodatku F). W szczególności, polecenia tego pliku umożliwiają zmianę standardowych (domyślnych) rozszerzeń nazw plików, zmianę katalogu z plikami źródłowymi, ustalenie kolorów "czarnych" i "białych" dla karty graficznej Hercules HGC itp.

5. EDYTOR WIERSZOWY

Edytor wierszowy umożliwia wprowadzenie i ewentualne poprawianie jednego wiersza tekstu programu. Edytor nie wymaga żadnego szczególnego wywoływania. Jego polecenia są zawsze dostępne po wywołaniu systemu programowania AC-LOGO. Dzięki temu jest on dobrym narzędziem do wprowadzania krótkich procedur. W celu wprowadzania i redagowania większych tekstów zaleca się korzystanie z edytora ekranowego, opisanego w następnym rozdziale.

Wprowadzanie znaków w edytorze wierszowym może odbywać się w dwóch trybach. W trybie "wstawianie" (por. rozdział 6) wprowadzany znak jest dopisywany w miejscu wskazywanym przez kursor po uprzednim rozsunięciu znaków od kursora do końca wiersza o jedną pozycję w prawo. W trybie "zastępowanie" wprowadzany znak zastępuje znak wskazywany przez kursor. Zmiana trybu następuje po naciśnięciu klawisza INS i jest sygnalizowana zmianą wysokości kursora.

Użytkownik dysponuje poleceniami przesuwania kursora oraz kasowania znaków. Kursor można przesuwać od początku wprowadzonego wiersza do ostatniego znaku w tym wierszu.

Polecenie przesuwania kursora:

↑ lub ↓	— o element zbioru poprzednich zawartości edytora,
← lub CTRL-S	— o jedną pozycję w lewo,
→ lub CTRL-D	— o jedną pozycję w prawo,
CTRL ← lub CTRL-A	— w lewo do początku słowa,
CTRL → lub CTRL-F	— w prawo do końca słowa,
HOME lub CTRL-R	— na początek wiersza,
END lub CTRL-C	— na koniec wiersza.

Wpisanie znaku CTRL-R bezpośrednio po przejściu do nowego wiersza powoduje skopiowanie poprzednio wprowadzonego wiersza. Wiersz ten może być następnie przeredagowany.

Polecenia kasowania tekstu:

CTRL-G lub DEL	— kasowanie znaku wskazywanego przez kursor,
BS	— kasowanie znaku znajdującego się na lewo od kursora,
^T	— kasowanie znaków od pozycji kursora w prawo, do końca słowa,
^Y	— kasowanie całego wiersza tekstu.

Inne polecenia:

CTRL-V lub INS	— zmiana trybu "wstawianie"/"zastępowanie".

Edytor wierszowy jest ściśle związany z interpretatorem poleceń kompilatora LOGO. Każdy wprowadzony wiersz tekstu jest traktowany jako ciąg wywoła procedur, przy czym niedomknięte listy są domykane bezpośrednio po wprowadzeniu znaku nowego wiersza CR, to znaczy liczba prawych nawiasów list jest uzupełniana do liczby lewych nawiasów. Tak utworzony ciąg wywołań procedur jest natychmiast wykonywany. Jedynym wyjątkiem od tej reguły jest wiersz, którego treścią jest wywołanie procedury pierwotnej OTO (ang. TO). Edytor wierszowy rozpoznaje to wywołanie i umożliwia zredagowanie całej procedury złożonej z więcej niż jednego wiersza tekstu przed przystąpieniem do jej analizy. Po rozpoznaniu wywołania procedury pierwotnej OTO, w kolejnym wierszu jest wyświetlany odmienny znak zgłoszenia — znak ">" zamiast znaku zapytania "?". Znak ten oznacza, że edytor wierszowy jest gotowy do przyjęcia kolejnego wiersza wprowadzanej procedury. Listy nie są tutaj automatycznie domykane. Zakończenie wprowadzania procedury następuje po rozpoznaniu przez edytor wierszowy wywołania procedury pierwotnej JUŻ (ang. END). Sterowanie jest wówczas przekazywane do kompilatora, który dokonuje analizy wpisanej procedury, tworzy jej postać półskompilowaną oraz drukuje komunikat o poprawnym lub błędnym wyniku kompilacji.

6. EDYTOR EKRANOWY

Edytor ekranowy służy do wprowadzania i poprawiania tekstów procedur. Jego nazwa pochodzi stąd, że kursor można w nim przesuwać po całym ekranie, nie tylko w jednym wierszu. Oprócz możliwości pionowego przemieszczania kursora różni się on od edytora wierszowego przede wszystkim operacjami przeszukiwania tekstu i operacjami blokowymi. Opisany w tym punkcie edytor ekranowy systemu AC-LOGO jest w większości funkcji zgodny z większością edytorów systemów zintegrowanych systemu operacyjnego MS-DOS (systemy TURBO firmy BORLAND INTERNATIONAL, SIDEKICK itp.).

Edytor ekranowy można wywołać przez wybór z głównego menu AC-LOGO, albo przez wywołanie procedur pierwotnych RED lub REDW (ang. EDIT i EDALL).

W celu wywołania edytora ekranowego z głównego menu systemu AC-LOGO można się posłużyć myszką lub klawiszami przesuwu kursora. Wywołanie edytora ekranowego za pomocą myszki następuje po przesunięciu kursora myszki na opcję EDYCJA i naciśnięciu lewego klawisza myszki. Podobnie, można wywołać edytor ekranowy z głównego menu systemu za pomocą klawiszy przesuwu kursora — należy przesunąć znacznik menu na opcję EDYCJA i nacisnąć klawisz ENTER. Szybką metodą wywołania edytora jest wywołanie przez naciśnięcie klawisza F3.

Po wywołaniu edytora z głównego menu systemu AC-LOGO w okienku edytora wyświetla się tekst ostatnio redagowanych procedur.

Wywołanie edytora z poziomu procedur LOGO ma dwie formy: z podaniem nazwy procedury, która ma być wprowadzana lub modyfikowana lub bez podania nazwy.

Pierwszą formę wywołania stosuje się w celu wprowadzenia zmian do jednej lub kilku wybranych procedur. W tej formie, edytor ekranowy wywołuje się przez wywołanie procedury pierwotnej RED (ang. EDIT). Jej parametrem może być słowo lub lista słów traktowanych jako nazwy redagowanych procedur. Jeżeli w wywołaniu podano nazwy istniejących procedur, to ich tekst zostanie wyświetlony na ekranie. Jeżeli nie ma procedury o nazwie podanej w wywołaniu, to część ekranu przeznaczona na tekst pozostanie pusta. Procedura RED może być wywoływana bezparametrowo. W tym przypadku parametrem domyślnym procedury RED jest nazwa lub nazwy ostatnio redagowanych procedur.

Drugą formę wywołania edytora ekranowego stosuje się w przypadku, gdy zamierzamy redagować wszystkie dotychczas zadeklarowane procedury. W takiej sytuacji należy wywołać bezparametrową procedurę pierwotną REDW (ang. EDALL).

Obydwie formy wywołania edytora umożliwiają zarówno poprawianie uprzednio zadeklarowanych procedur, jak i wprowadzanie nowych. Inaczej mówiąc nazwa wprowadzanej procedury nie musi być znana w momencie wywoływania edytora.

6.1. Elementy informacyjne okna edytora

Po zgłoszeniu się edytora na ekranie zostaje wyświetlone okno edytora przeznaczone na tekst programu. Okno może być puste, jeżeli rozpoczynamy wprowadzanie nowego tekstu lub może zawierać tekst wcześniej wprowadzony, jeżeli chcemy go modyfikować. Okno edytora może zajmować cały ekran lub jego część. Okno jest wyświetlane w ramce. Jeżeli okno zajmuje cały ekran, to jest wyświetlane bez wiersza podpowiedzi i bez głównego menu systemu okienkowego. W pierwszym od góry wierszu ekranu jest wtedy wyświetlany wiersz głównego menu systemu AC-LOGO, a w ostatnim — wiersz podpowiedzi. W ramce edytora ekranowego, podobnie jak w ramkach innych okienek, znajdują się pola informujące o rozmiarach i pozycji wyświetlanego w okienku fragmentu tekstu, pola umożliwiające zmianę rozmiaru i pozycji okienka za pomocą myszki oraz dodatkowo pola informujące o pozycji kursora tekstowego. Pierwsze z nich zawiera numer wiersza, a drugie — numer kolumny, w której aktualnie znajduje się kursor tekstowy.

Wiersz głównego menu służy do szybkiego wywoływania funkcji systemu AC-LOGO. Przejście z edytora do głównego menu następuje po naciśnięciu klawisza ESC, prawego klawisza myszki lub po dokonaniu wyboru jednej z opcji menu za pomocą myszki i naciśnięciu lewego klawisza myszki. Naciśnięcie klawisza ESC lub prawego klawisza myszki podczas pracy w okienku edytora zajmującym cały ekran powoduje wyświetlenie głównego menu i przejście do wyboru jego opcji.

Wiersz podpowiedzi edytora zawiera informacje o funkcjach klawiszy w danym kontekście.

6.2. Kursory

W czasie pracy edytora, na ekranie są wyświetlane dwa kursory: kursor myszki i kursor tekstowy.

Kursor myszki, widoczny tylko wtedy, gdy myszka jest poprawnie zainstalowana, ma postać strzałki. Kursor ten można przesuwać po całym

ekranie, za pomocą ruchów myszką. Kursor myszki służy zarówno do wskazywania miejsc tekstu, w których mają być wykonane operacje edytora, jak i do wykonywania operacji związanych ze środowiskiem edytora, takich jak zmiana rozmiaru okna, przejście do głównego menu LOGO lub wywołanie kontekstowego systemu podpowiedzi. Lewy klawisz myszki służy do potwierdzania wyboru operacji lub pozycji kursora myszki, prawy klawisz do przejścia do głównego menu, a środkowy do wywoływania kontekstowego systemu podpowiedzi. Wybranie odpowiedniego fragmentu tekstu kontekstowego systemu podpowiedzi polega na wskazaniu kursorem myszki wybranej opcji menu lub słowa w oknie edytora i naciśnięciu środkowego klawisza myszki. Na ekranie zostanie wyświetlone okienko z odpowiednim tekstem podpowiedzi.

Kursor tekstowy jest wyświetlany wewnątrz okienka edytora. Kursor tekstowy wskazuje miejsce tekstu, w którym są dokonywane zmiany. Kursor tekstowy ma dwie postaci: dolnej połowy znaku wyświetlonej w negacji lub całego znaku wyświetlonego w negacji. Postać kursora tekstowego informuje o trybie pracy edytora. Pierwsza postać oznacza, że edytor pracuje w trybie "Wstawianie", w którym wpisanie znaku w dowolnym miejscu istniejącego tekstu powoduje przesunięcie wszystkich znaków od kursora do końca wiersza o jedną pozycję w prawo i wprowadzenie wpisanego znaku na tak utworzoną wolną pozycję, czyli w miejscu wskazywanym uprzednio przez kursor. Druga postać kursora oznacza, że edytor pracuje w trybie "Zamiana", w którym wpisanie dowolnego znaku w miejscu istniejącego znaku powoduje skasowanie poprzedniego znaku i wstawienie w to miejsce nowego.

Kursor tekstowy może wskazywać słowa podczas wywoływania kontekstowego systemu podpowiedzi. Wybranie odpowiedniego fragmentu tekstu kontekstowego systemu podpowiedzi z użyciem kursora tekstowego polega na wskazaniu tym kursorem wybranego słowa w oknie edytora, przytrzymaniu klawisza CTRL i naciśnięciu klawisza F1. Na ekranie zostanie wyświetlone okienko z odpowiednim tekstem podpowiedzi.

6.3. Przesuwanie kursora tekstowego

W dowolnej chwili pracy edytora można zmienić położenie kursora. Pole, w którym można przemieszczać kursor, jest ograniczone od lewej strony pierwszą kolumną tekstu, od prawej strony tysiąc dwudziestą czwartą kolumną tekstu, od góry początkiem tekstu, a od dołu końcem tekstu. Maksymalny rozmiar tekstu to 64500 znaków.

Przesuwania kursora dokonuje się za pomocą klawiszy ze strzałkami, umieszczonych z prawej strony klawiatury lub za pomocą znaków sterujących. Jeśli w systemie jest zainstalowana myszka, to przesuwanie kursora

można realizować także za pomocą myszki. W tym celu należy przesunąć kursor myszki w wybrane miejsce, a następnie nacisnąć i zwolnić lewy klawisz myszki (por. rozdział 7).

Kursor można przesuwać poziomo i pionowo. W kierunku poziomym kursor może być przesuwany o jeden znak, jedno słowo oraz do początku i do końca tekstu w wierszu. W edytorze słowo jest rozumiane szerzej niż w kompilatorze — dla edytora słowem jest dowolny ciąg znaków oddzielony separatorami, którymi są znaki: #$^()+*{}:[]/@,-<> oraz znak spacji, tabulacji i nowej linii. W kierunku pionowym kursor może być przesuwany o jeden wiersz, o jeden ekran oraz do początku i końca tekstu. Możliwości te są realizowane przez trzy grupy poleceń. Pierwsza grupa obejmuje polecenia przesuwania kursora poziomo o jeden znak i pionowo o jeden wiersz, druga grupa — poziomo o jedno słowo i pionowo o jeden ekran, a trzecia grupa poziomo do początku i końca wiersza oraz pionowo do początku i końca ekranu oraz początku i końca tekstu.

Polecenia pierwszej grupy są wydawane w następujący sposób:

- przesunięcie kursora o jedną pozycję w lewo następuje po wciśnięciu klawisza oznaczonego strzałką w lewo lub po wpisaniu znaku ^S;
- przesunięcie kursora o jedną pozycję w prawo następuje po wciśnięciu klawisza oznaczonego strzałką w prawo lub po wpisaniu znaku ^D;
- przesunięcie kursora o jeden wiersz w górę następuje po wciśnięciu klawisza oznaczonego strzałką w górę lub po wpisaniu znaku ^E;
- przesunięcie kursora o jeden wiersz w dół następuje po wciśnięciu klawisza oznaczonego strzałką w dół lub po wpisaniu znaku ^X.

Polecenia drugiej grupy są wydawane następująco:

- przesunięcie kursora w lewo, do początku słowa następuje po wciśnięciu klawisza CTRL i jednocześnie klawisza oznaczonego strzałką w lewo lub po wpisaniu znaku ^A;
- przesunięcie kursora w prawo, do początku następnego słowa następuje po wciśnięciu klawisza CTRL i jednocześnie klawisza oznaczonego strzałką w prawo lub po wpisaniu znaku ^F;
- przesunięcie kursora w górę o jeden ekran następuje po wciśnięciu klawisza CTRL i jednocześnie klawisza oznaczonego strzałką w górę lub po wpisaniu znaku ^R. W przypadku, gdy kursor znajduje się w wierszu o numerze mniejszym niż 23, to nastąpi przesunięcie do pierwszego wiersza tekstu;
- przesunięcie kursora w dół o jeden ekran następuje po wciśnięciu klawisza CTRL i jednocześnie klawisza oznaczonego strzałką w dół lub po wpisaniu znaku ^C. W przypadku, gdy kursor znajduje się w pozycji oddalonej o mniej niż 23 wiersze od końca tekstu, nastąpi przesunięcie do ostatniego wiersza tekstu.

Trzecia grupa poleceń obejmuje:
- przesunięcie kursora do początku wiersza po wydaniu polecenia ^QS[1] lub znaku ^Q i następnie znaku strzałki w lewo;
- przesunięcie kursora do końca wiersza po wydaniu polecenia ^QD lub znaku ^Q i następnie znaku strzałki w prawo;
- przesunięcie kursora do początku ekranu po wydaniu polecenia ^QE lub znaku ^Q i następnie znaku strzałki w górę;
- przesunięcie kursora do końca ekranu po wydaniu polecenia ^QX lub znaku ^Q i następnie znaku strzałki w dół;
- przesunięcie kursora do początku tekstu po wydaniu polecenia ^QR;
- przesunięcie kursora do końca tekstu po wydaniu polecenia ^QC;
- przesunięcie kursora do wybranego wiersza tekstu za pomocą polecenia ^OG. Po wydaniu tego polecenia na ekranie jest wyświetlane okienko, w którym należy wpisać numer wybranego wiersza tekstu.

Oprócz przesuwania kursora po tekście, można również przesuwać tekst w oknie edytora, bez zmiany położenia kursora w tekście. Wydanie polecenia ^W powoduje przesunięcie całego tekstu umieszczonego w oknie edytora o jeden wiersz w dół, natomiast polecenia ^Z — o jeden wiersz w górę. W obu przypadkach kursor jest przesuwany wraz z tekstem. Wyjątkiem od tej reguły jest sytuacja, w której kursor znajduje się w skrajnym wierszu okna edytora. W takim przypadku kursor pozostaje w nim.

6.4. Kasowanie tekstu

Edytor oferuje pięć możliwości kasowania tekstu:
- kasowanie znaku wskazywanego przez kursor po naciśnięciu klawisza DEL lub wpisaniu znaku ^G,
- kasowanie znaku znajdującego się na lewo od kursora po wciśnięciu klawisza BS,
- kasowanie znaków od pozycji kursora w prawo, do końca słowa po wpisaniu znaku ^T,
- kasowanie znaków od pozycji kursora w prawo, do końca wiersza po wydaniu polecenia ^QY,
- kasowanie wiersza, w którym znajduje się kursor po wpisaniu znaku ^Y.

---

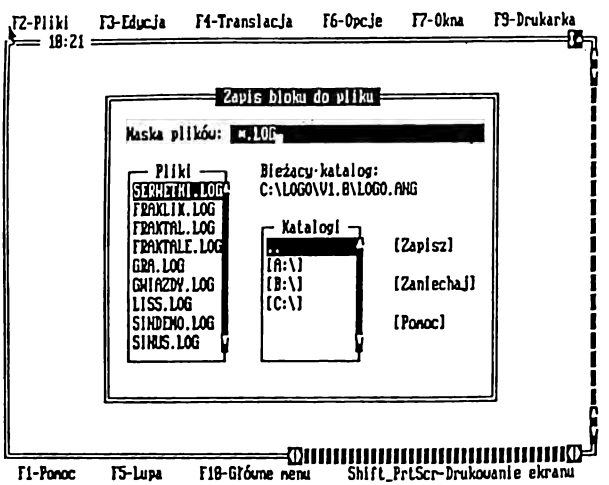
[1] Zapis ten oznacza, że w celu wydania polecenia należy wcisnąć kombinację klawiszy CTRL i Q, a następnie wcisnąć klawisz S lub kombinację klawiszy S i SHIFT albo S i CTRL

6.5. Operacje na bloku tekstu

Edytor ekranowy AC-LOGO pozwala na wykonywanie operacji na fragmentach tekstu zwanych *blokami*. Programista może zaznaczyć blok o dowolnej wielkości. W szczególności blokiem może być jeden znak i cały tekst.

W celu zaznaczenia bloku należy ustalić jego początek i koniec. Początek bloku zaznaczamy przez umieszczenia kursora na pierwszym znaku, który ma wejść w skład bloku, i wydanie polecenia ^KB. Koniec bloku zaznaczamy przez umieszczenie kursora na ostatnim znaku, który ma wejść w skład bloku, i wydanie polecenia ^KK. Jeżeli blokiem ma być tylko jedno słowo, to wystarczy umieścić kursor na dowolnym znaku tego słowa i wydać polecenie ^KT. Blok można również zaznaczyć posługując się myszką. W tym celu należy wskazać kursorem myszki początek (lub koniec) bloku, nacisnąć lewy klawisz myszki, nie zwalniając go przesunąć kursor myszki na koniec (początek) bloku i zwolnić lewy klawisz myszki.

Blok jest oznaczony na ekranie przez wyświetlenie w negacji i tylko wówczas można wykonywać na nim operacje. Polecenie ^KH pozwala na zlikwidowanie efektu wyświetlania bloku w negacji, bez zmiany jego położenia. Ponowne wydanie polecenia ^KH powoduje przywrócenie efektu wyświetlania w negacji.



Rys. 6.1. Okienko wyboru pliku przy zapisie bloku

Na blokach można wykonywać następujące operacje:

- kopiowanie,
- przesuwanie,
- kasowanie,
- zapis do pliku,
- odczyt z pliku.

W celu skopiowania bloku należy ustawić kursor w miejscu, w którym ma się znaleźć początek bloku po skopiowaniu i wydać polecenie ^KC.

W celu przesunięcia bloku należy ustawić kursor w miejscu, w którym ma się znaleźć początek bloku po przesunięciu i wydać polecenie ^KV.

W celu skasowania zaznaczonego bloku należy wydać polecenie ^KY.

W celu zapisania zaznaczonego bloku w pliku należy wydać polecenie ^KW. Na ekranie zostanie wyświetlone okienko wyboru pliku (rys. 6.1).

Okienko to składa się z następujących elementów:

- pola "podaj maskę plików", w którym wpisujemy nazwę pliku lub maskę plików zgodną z regułami systemu DOS, powodującą selekcję plików wyświetlanych w podokienku "pliki". Standardową maską jest maska *.LOG;
- pola "Bieżący katalog", w którym jest wyświetlana bieżąca ścieżka dostępu do plików,
- podokienka "Pliki", w którym są wyświetlanę nazwy plików zgodne z wybraną maską;
- podokienka "Katalogi", w którym są wyświetlane nazwy dysków ujęte w nawiasy kwadratowe i nazwy katalogów dostępnych z katalogu bieżącego;
- pola "Zapisz", którego wybranie powoduje wykonanie zapisu;
- pola "Przerwij", którego wybranie powoduje powrót do edycji bez zapisu;
- pola "Pomoc", którego wybranie powoduje wywołanie systemu pomocy.

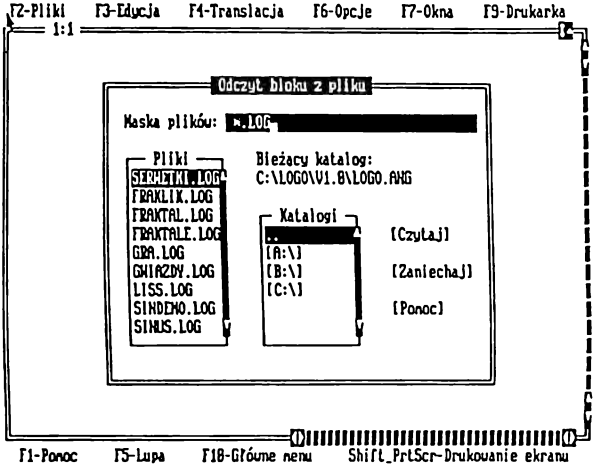
Przejścia z pola do pola wykonuje się za pomocą klawiszy TAB i SHIFT-TAB (por. roz. 7).

W celu wyboru pliku należy wpisać nazwę pliku w polu "Maska plików" i nacisnąć klawisz ENTER, lub wybrać w okienku "Pliki" jeden z wyświetlonych plików i nacisnąć ENTER lub nacisnąć lewy klawisz myszki, wskazując na pole "Zapisz" (por. roz. 7). Jeżeli plik znajduje się w innym katalogu niż bieżący, to należy za pomocą klawiszy TAB lub SHIFT-TAB przejść do podokienka "Katalogi" i zmienić bieżący katalog. Następnie należy za pomocą klawiszy TAB lub SHIFT-TAB przejść do podokienka "Pliki", wybrać plik i nacisnąć ENTER. Jeżeli plik do którego chcemy zapisać już istnieje, to poprzednia wartość pliku zostanie zapisana w pliku o tej samej nazwie lecz o rozszerzeniu .BAK.

W celu odczytania zawartości pliku do bloku tekstu należy wydać polecenie ^KR. Na ekranie zostanie wyświetlone okienko wyboru pliku

(rys. 6.2). Tak jak przy zapisie bloku, należy wpisać lub wybrać nazwę pliku. Po wczytaniu blok zostanie podświetlony.

Korzystanie z operacji na bloku tekstu ułatwiają polecenia przesuwania kursora do początku i końca bloku. Polecenie ^QB powoduje umieszczenie kursora na pierwszym znaku bloku, natomiast polecenie ^QK powoduje umieszczenie kursora na ostatnim znaku bloku.



Rys. 6.2. Okienko wyboru pliku przy odczycie bloku

6.6. Przeszukiwanie i zamiana tekstu

Edytor ekranowy pozwala na wyszukiwanie w przetwarzanym tekście wskazanych sekwencji znaków. Wyszukiwanie może być połączone z zamianą znalezionych sekwencji znaków na inne.

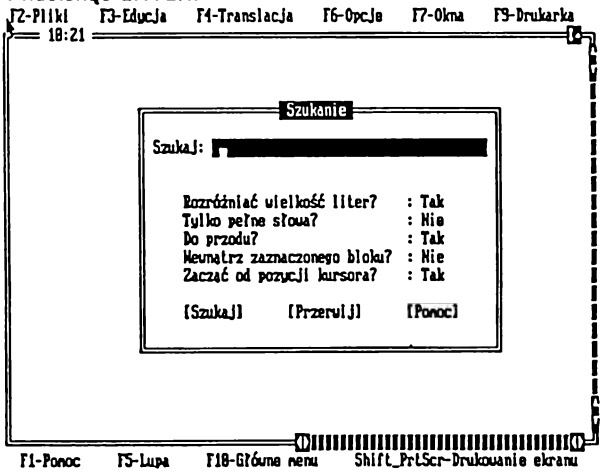
W celu wyszukania w tekście określonej sekwencji znaków należy wydać polecenie ^QF. Spowoduje to wyświetlenie okienka "Szukanie" (rys. 6.3). W pierwszym polu tego okienka należy wpisać szukaną sekwencję znaków i nacisnąć ENTER. Jako domyślne edytor podpowiada w tym polu słowo, w którym znajduje się kursor. Kolejne pola tego okienka zawierają opcje szukania. Przejście z jednej opcji do drugiej następuje po naciśnięciu klawiszy przesuwu kursora, TAB lub SHIFT-TAB. Zmiana wartości opcji następuje po naciśnięciu klawisza ENTER. Są dostępne następujące opcje:

- "Rozróżniać wielkość liter" (Tak/Nie); na przykład po przyjęciu za wzór sekwencji znaków "oto", gdy opcja ma wartość "Tak", zostanie znalezione zarówno słowo "oto", jak i "Oto" oraz "OTO",

- "Tylko pełne słowa" (Tak/Nie); na przykład po przyjęciu za wzór sekwencji znaków "oto", gdy opcja ma wartość "Tak", słowo "oto" zostanie znalezione, a słowo "otok" nie zostanie znalezione,
- "Do przodu" (Tak/Nie); wybranie wartości "Tak" powoduje przeszukiwanie w kierunku końca tekstu, a "Nie" — początku;
- "Wewnątrz zaznaczonego bloku" (Tak/Nie); wybranie wartości "Tak" powoduje przeszukiwanie tylko w bloku tekstu, a "Nie" — w całym tekście;
- "Zacząć od pozycji kursora" (Tak/Nie); wybranie wartości "Tak" powoduje rozpoczęcie przeszukiwania w miejscu, w którym znajduje się kursor tekstowy, a "Nie" — w początku zakresu przeszukiwania; w zależności od wartości opcji "Do przodu" i "Wewnątrz zaznaczonego bloku" początkiem zakresu jest:

Do przodu	Wewnątrz...	
Nie	Nie	— koniec tekstu
Nie	Tak	— koniec bloku
Tak	Nie	— początek tekstu
Tak	Tak	— początek bloku

Po ustaleniu wartości wszystkich opcji należy przejść kursorem do pola [Szukaj] i nacisnąć ENTER.



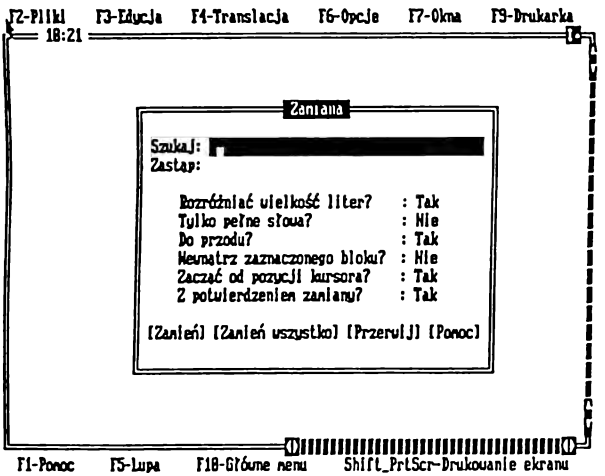
Rys. 6.3. Okienko "Szukanie"

Jeżeli sekwencja znaków podana jako wzór zostanie odszukana, to kursor zatrzyma się za ostatnim znakiem tej sekwencji. Natomiast jeżeli w przeszukiwanym tekście nie ma sekwencji znaków podanej jako wzór, to zostanie wyświetlone okienko z napisem:

Nie znaleziono poszukiwanej frazy

W takim przypadku, przed wydaniem kolejnych poleceń należy wcisnąć klawisz ESC.

W celu zamiany pewnej sekwencji znaków na inną należy wydać polecenie ^QA. Spowoduje to wyświetlenie okienka "Zamiana" (rys. 6.4). W pierwszym polu tego okienka należy wpisać szukaną sekwencję znaków i nacisnąć ENTER. Jako domyślne edytor podpowiada w tym polu słowo, w którym znajduje się kursor. W drugim polu okienka należy wpisać sekwencję, na którą ma być zamieniona sekwencja podana w pierwszym polu okienka. Po naciśnięciu klawisza ENTER kursor przechodzi do pierwszego pola opcji zamiany. Pierwsze pięć opcji ma takie samo znaczenie jak przy szukaniu frazy. Ostatnia opcja "Z potwierdzeniem zamiany" (Tak/Nie) określa, czy po znalezieniu frazy, przed zamianą ma być wyświetlane okienko z zapytaniem "Zamienić? Tak/Nie", czy zamiana ma zostać wykonana bez pytania o zgodę.



Rys. 6.4. Okienko "Zamiana"

Po ustaleniu wartości wszystkich opcji należy przejść kursorem do pola [Zastąp] lub do pola [Zastąp wszystko] i nacisnąć ENTER. Wybranie pierwszego z wyżej wymienionych pól spowoduje wyszukanie i zamianę pierwszego napotkanego wystąpienia szukanej frazy, a wybranie [Zastąp wszystko] — wszystkich wystąpień frazy wewnątrz określonego przez opcje zakresu szukania.

Jeżeli w przeszukiwanym tekście nie ma sekwencji znaków podanej jako wzór, to zostanie wyświetlone okienko z napisem:

Nie znaleziono poszukiwanej frazy

W takim przypadku, przed wydaniem kolejnych poleceń należy wcisnąć klawisz ESC.

Ostatnio wykonaną operację szukania lub zamiany z tym samym wzorem i opcjami można powtórzyć wydając polecenie ^L. Jest ono bardzo przydatne przy przeszukiwaniu tekstu krok po kroku.

6.7. Inne operacje

Oprócz operacji opisanych powyżej edytor ekranowy oferuje użytkownikowi następujące dalsze funkcje:

- umieszczanie w tekście znaczników,
- skok do wybranego znacznika,
- wprowadzenie do tekstu znaku nowego wiersza,
- odtworzenie poprzedniego stanu przetwarzanego wiersza,
- powrót kursora do poprzedniej pozycji,
- zmianę trybów pracy edytora,
- przerwanie wykonywania polecenia.

Do ułatwienia szybkiego znajdowania wybranych miejsc w tekście służą znaczniki. W celu umieszczenia znacznika w miejscu wskazywanym przez kursor należy wydać jedno z poleceń: ^K1, ^K2, .. ^K9. W celu przesunięcia kursora do wybranego znacznika należy wydać jedno z poleceń: ^Q1, ^Q2, .. ^Q9.

Wprowadzenie do tekstu znaku nowego wiersza wymaga wydania polecenia ^N. Po wydaniu tego polecenia kursor nie zmienia swego położenia, natomiast bezpośrednio za nim zostaje w tekście umieszczony znak nowego wiersza. Podobny efekt uzyskuje się po wciśnięciu klawisza ENTER. Różnica polega na tym, że po wciśnięciu klawisza ENTER kursor przechodzi do nowego wiersza.

Odtworzenie stanu przetwarzanego wiersza następuje po wydaniu polecenia ^QL. Po umieszczeniu kursora w danym wierszu jego stan jest zapamiętywany aż do momentu przesunięcia kursora do innego wiersza. Jeżeli w wierszu wprowadzono zmiany (na przykład dopisanie lub skasowanie tekstu), to przed przesunięciem kursora do innego wiersza można odtworzyć jego stan pierwotny. Po przesunięciu kursora takie odtworzenie nie jest już możliwe.

Przesunięcie kursora do poprzednio zajmowanej pozycji następuje po wydaniu polecenia ^QP. Polecenie to jest najczęściej stosowane po poleceniach przeszukiwania lub zamiany tekstu.

Wyboru trybu pracy edytora: "Wstawianie"/"Zamiana" dokonuje się przez naciśnięcie klawisza INS lub poleceniem ^V. Obie metody zmieniają tryb bieżący na przeciwny.

Włączenie trybu tabulacji dokonuje się poleceniem ^QI; to samo polecenie wydane powtórnie wyłącza ten tryb.

Przerwanie wykonywania polecenia następuje po wydaniu polecenia ^U. Polecenie ^U można wydać w trzech przypadkach:

- po wprowadzeniu pierwszego znaku polecenia dwuznakowego,
- w czasie, gdy edytor czeka na wprowadzenie parametru jakiegoś polecenia,
- w czasie szukania lub zmiany.

6.8. Zakończenie edycji

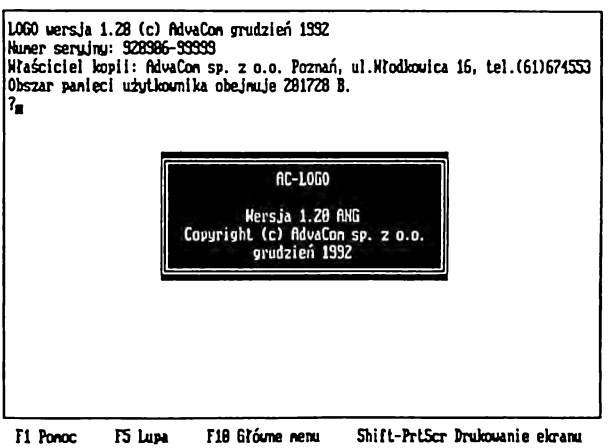
W celu zakończenia edycji należy wydać polecenie ^KD. Polecenie to można wydać w dowolnej chwili pracy edytora, bez względu na to, w którym miejscu tekstu jest umieszczony kursor. Po wydaniu tego polecenia praca edytora zostaje zakończona, jego okienko zostaje zamknięte i kompilator rozpoczyna analizę wszystkich procedur, które były redagowane. W przypadku stwierdzenia błędu w jakiejś procedurze, na ekranie jest wyświetlany komunikat mówiący o typie błędu. Po zakończeniu kompilacji następuje przejście do edytora wierszowego.

Oprócz zakończenia edycji, opisanego powyżej, można również zaniechać edycji rezygnując z wszystkich zmian dokonanych od chwili ostatniego wywołania edytora. Dokonuje się tego poleceniem ^KQ.

7. SYSTEM OKIENKOWY

7.1. Ogólne zasady poruszania się w systemie menu

Wywołanie zintegrowanego systemu LOGO następuje w wyniku wykonania polecenia LOGO. Bezpośrednio po realizacji tej czynności, na ekranie pojawia się główne menu a na jego tle informacja o numerze wersji systemu:



Naciśnięcie dowolnego klawisza lub odczekanie około pięciu sekund powoduje usunięcie tej informacji i odsłonięcie ekranu podzielonego na następujące części:

- wiersz głównego menu,
- okno graficzno-tekstowe z zaznaczonym znakiem zachęty i kursorem,
- wiersz klawiszy pomocniczych.

Należy zaznaczyć, że zawartość ekranu po uruchomieniu systemu LOGO jest zależna od treści pliku konfiguracyjnego. Przedstawiony powyżej podział ekranu został przyjęty jako domyślny (tzn. na wypadek braku opisu podziału

ekranu w pliku konfiguracyjnym). Dokładny opis poleceń z pliku konfiguracyjnego systemu LOGO zawarto w dodatku F.

W każdej chwili pracy systemu aktywne jest główne menu lub któreś z okien: graficzno-tekstowe lub edytora ekranowego. Ponadto na ekran mogą być przywoływane okienka pomocnicze związane z realizacją odpowiednich funkcji systemu (np. z odczytem lub zapisem pliku na dysk itp.).

Bezpośrednio po uruchomieniu systemu LOGO aktywne jest okno graficzno-tekstowe, obsługiwane między innymi przez edytor wierszowy.

Wyróżnienie wiersza głównego menu następuje po naciśnięciu klawisza F10 lub ESC. Objawia się to podświetleniem jednej z jego opcji.

Wybranie dowolnej opcji głównego menu może być dokonane na kilka sposobów. Najprostszym z nich jest naciśnięcie odpowiedniego klawisza funkcyjnego związanego z daną opcją. Innym sposobem jest naciśnięcie klawisza F10 lub ESC, podświetlenie wybranej opcji za pomocą klawiszy sterujących ruchem kursora "←" lub "→" i naciśnięcie klawisza ENTER. W przypadku większości opcji głównego menu (za wyjątkiem opcji Edycja) nastąpi wtedy wywołanie pod-menu. Wybrania opcji pod-menu można dokonać przez wprowadzenie z klawiatury znaku wyróżnionego w nazwie danej opcji lub przez użycie klawiszy "↑" i "↓" i naciśnięcie klawisza ENTER (co z kolei może spowodować pojawienie się kolejnego pod-menu itd.). Znak wywołania opcji jest wyróżniony w nazwie opcji przez podkreślenie. Z każdego pod-menu można cofnąć się na wyższy poziom posługując się klawiszem ESC. Naciśnięcie tego klawisza w głównym menu powoduje uaktywnienie okienka graficzno-tekstowego.

Należy dodać, że jeśli po wybraniu pierwszego pod-menu zostaną użyte klawisze "→" i "←", to na ekran będą przywoływane sąsiednie pod-menu głównego menu. Właściwość ta jest bardzo przydatna w początkowym okresie zapoznawania się z systemem, kiedy rozkład opcji między poszczególne pod-menu nie jest dobrze znany.

Wyboru każdej opcji menu można dokonać również przy pomocy myszki. Lewy przycisk myszki odpowiada klawiszowi ENTER, a prawy klawiszowi ESC. W celu wybrania odpowiedniej opcji należy wskazać kursorem myszki którąkolwiek z liter w nazwie opcji i nacisnąć lewy przycisk. Powrót na wyższy poziom menu następuje po naciśnięciu prawego przycisku myszki.

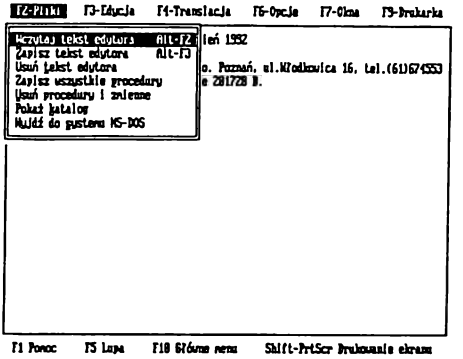
Środkowy klawisz myszki odpowiada wciśnięciu klawisza F1 — jest to wywołanie systemu podpowiedzi. Należy zauważyć, że dla myszki typu "MicroSoft MOUSE", która posiada tylko dwa klawisze, nie jest możliwe wywołanie systemu podpowiedzi przy pomocy myszki. Możliwa jest także sytuacja, że trzyklawiszowa myszka pracująca w trybie "MicroSoft MOUSE" nie będzie umożliwiała wykorzystania środkowego klawisza. W takim przypadku należy uaktywnić taki program obsługi myszki, który udostępnia wszystkie trzy klawisze.

7.2. Opis opcji głównego menu

W skład głównego menu systemu LOGO wchodzą następujące opcje: "Pliki", "Edycja", "Translacja", "Opcje", "Okna", "Drukarka".

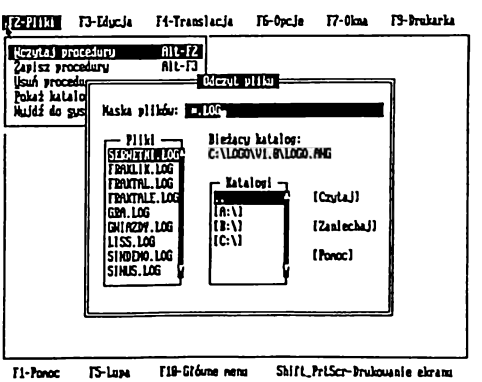
Opcja Pliki

Opcję "Pliki" wybiera się przez naciśnięcie klawisza F2. Po jej uaktywnieniu na ekranie pojawia się następujące pod-menu:



Opcja "Wczytaj tekst edytora"

Opcja ta umożliwia wczytanie pliku tekstowego do edytora ekranowego. Po jej wybraniu, na ekranie pojawia się następujące okienko:



Okienko to składa się z następujących elementów:

- pola "Maska plików", w którym wpisujemy nazwę pliku lub maskę plików zgodną z regułami systemu DOS, powodującą selekcję plików wyświetlanych w podokienku "Pliki". Standardową maską jest maska *.LOG;
- pola "Bieżący katalog", w którym jest wyświetlana bieżąca ścieżka dostępu do plików,
- podokienka "Pliki", w którym są wyświetlane nazwy plików zgodne z wybraną maską;
- podokienka "Katalogi", w którym są wyświetlane nazwy dysków ujęte w nawiasy kwadratowe i nazwy katalogów dostępnych z katalogu bieżącego;
- pola "Czytaj", którego wybranie powoduje wczytanie pliku do edytora;
- pola "Przerwij", którego wybranie powoduje powrót do edycji bez zapisu;
- pola "Pomoc", którego wybranie powoduje wywołanie systemu pomocy.

Przejścia z pola do pola wykonuje się za pomocą klawiszy TAB i SHIFT-TAB.

Wyboru pliku można dokonać na kilka sposobów. Pierwszym sposobem jest wpisanie nazwy pliku w polu "Maska plików" i naciśnięcie klawisza ENTER. W drugim sposobie należy wpisać w polu "Maska plików" nazwę niejednoznaczną, zawierającą znaki * i ? i nacisnąć ENTER. W okienku "pliki" są następnie wyświetlane nazwy plików zgodne z wpisaną maską. Należy wybrać jedną z wyświetlonych nazw i nacisnąć ENTER lub nacisnąć lewy klawisz myszki, wskazując na pole "Czytaj". Jeżeli plik znajduje się w innym katalogu lub na innym dysku niż bieżący, to należy za pomocą klawiszy TAB lub SHIFT-TAB przejść do podokienka "Katalogi" i zmienić bieżący katalog lub dysk. Następnie należy za pomocą klawiszy TAB lub SHIFT-TAB przejść do podokienka "Pliki", wybrać plik i nacisnąć ENTER. Zmianę katalogu można wykonać za pomocą myszy, wskazując kursorem myszy żądany katalog w okienku katalogu, a następnie naciskając jej lewy klawisz.

Zaniechania realizacji powyższej opcji można dokonać przez naciśnięcie klawisza ESC lub prawego przycisku myszy.

Opcja "Zapisz tekst edytora"

Opcja ta umożliwia zapisanie na dysku tekstu procedur znajdujących się w edytorze ekranowym. Po jej wybraniu na ekranie pojawia się podobne okienko jak w przypadku wybrania opcji "Wczytaj procedury". Okienko te różni się od poprzedniego tylko nagłówkami i tym, że zamiast pola "Czytaj" ma pole "Zapisz". Określenia nazwy pliku do zapisu tekstu znajdującego się w edytorze ekranowym dokonuje się w ten sam spoób, jak w przypadku poprzedniej opcji. Jeżeli plik do którego chcemy zapisać już istnieje, to

poprzednia wartość pliku zostanie zapisana w pliku o tej samej nazwie lecz o rozszerzeniu BAK.

Opcja "Zapisz wszystkie procedury"

Opcja ta umożliwia zapisanie na dysku tekstów wszystkich zdefiniowanych procedur. Po jej wybraniu na ekranie pojawia się podobne okienko jak w przypadku wybrania opcji "Zapisz tekst edytora".

Opcja "Usuń tekst edytora"

Wybranie tej opcji powoduje usunięcie tekstu znajdującego się aktualnie w edytorze ekranowym. Umożliwia to rozpoczęcie redagowania kolejnego tekstu.

Opcja "Usuń wszystkie procedury"

Wybranie tej opcji powoduje usunięcie tekstów wszystkich zdefiniowanych procedur i zmiennych LOGO. Umożliwia to odzyskanie zajętej przez te procedury i zmienne pamięci.

Opcja "Pokaż katalog"

Opcja ta umożliwia obejrzenie zawartości żądanego katalogu. Po jej wybraniu, na ekranie pojawia się okienko z wydrukowaną zawartością wybranego katalogu. Operacje dostępne w okienku są prawie identyczne jak operacje opisane dla opcji "Wczytaj procedury". Jedyną różnicą jest to, że w przypadku podświetlenia nazwy pliku i naciśnięcia klawisza ENTER nie zostanie podjęta żadna akcja. W przypadku podświetlenia nazwy katalogu i naciśnięcia klawisza ENTER, w górnej ramce pojawi się nazwa wybranego katalogu, a w dolnej jego zawartość.

Opcja "Wyjdź do Systemu MS-DOS"

Opcja ta powoduje zakończenie działania systemu AC-LOGO i przekazanie sterowania do systemu operacyjnego. Przed wykonaniem tej operacji system AC-LOGO upewnia się zadając pytanie:

Jesteś pewien (T/N)?

Jeśli w edytorze ekranowym znajduje się tekst, który od czasu ostatniego zapisu został zmodyfikowany, na ekranie pojawia się okno z pytaniem:

Tekst edytora nie został zachowany. Koniec pracy?

Wybranie odpowiedzi "Tak" powoduje zakończenie pracy AC-LOGO, a odpowiedzi "Nie" — przejście do okienka graficzno-tekstowego.

Opcja "Edycja"

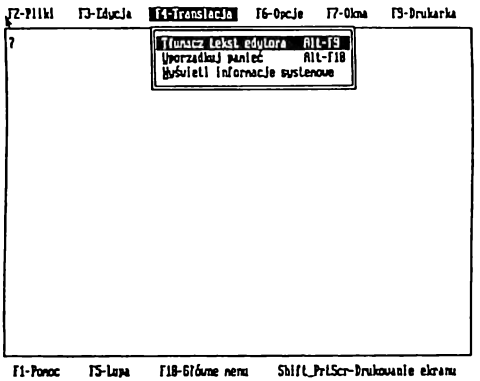
Opcja ta powoduje otwarcie okna edytora ekranowego i umożliwia tym samym rozpoczęcie lub kontynuowanie redagowania tekstu. Wybiera się ją przez naciśnięcie klawisza F3.

Okno edytora ekranowego opisano w rozdziale 6.

Wielkość okna edytora i jego położenie można zmieniać, korzystając z opcji "Okna", lub klawisza pomocniczego Ctrl-F4, bądź też za pomocą myszy ciągnąc ramkę okna przy naciśniętym lewym przycisku myszy.

Opcja "Translacja"

Opcję "Translacja" wybiera się przez naciśnięcie klawisza F4. Po jej uaktywnieniu na ekranie pojawia się następujące pod-menu:



Opcja "Tłumacz tekst edytora"

Opcja ta powoduje rozpoczęcie procesu translacji tekstu zawartego w edytorze ekranowym. Można ją wywołać przez jednoczesne naciśnięcie klawisza ALT i klawisza F9. Bezpośrednio po jej wywołaniu, na ekranie pojawia się okienko, w którym są wyświetlane nazwy tłumaczonych procedur. Po zapełnieniu okienka jest ono przewijane.

Jeśli proces tłumaczenia zakończy się powodzeniem, nastąpi automatyczne przejście do edytora wierszowego. Jeśli podczas tłumaczenia wystąpi błąd, w okienku zostanie wyświetlony odpowiedni komunikat. Edytor ekranowy można w takiej sytuacji wywołać w sposób standardowy, na przykład przez naciśnięcie klawisza F3.

Opcja "Wyświetl informacje systemowe"

Opcja ta umożliwia użytkownikowi zorientowanie się w aktualnej zajętości pamięci operacyjnej systemu LOGO. Po wybraniu tej opcji, na ekranie pojawia się okienko, opisujące aktualny podział pamięci operacyjnej między poszczególne bloki.

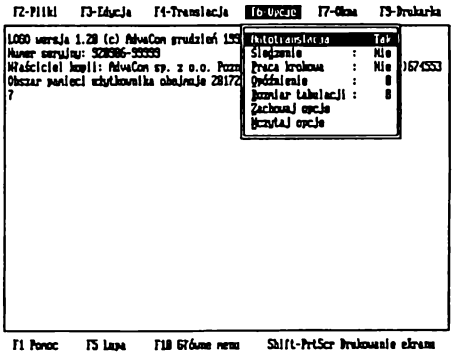
Powrotu na wyższy poziom można dokonać przez naciśnięcie klawisza ESC, prawego przycisku myszki, lub wybranie pola [Gotowe].

Opcja "Uporządkuj pamięć"

Opcja ta umożliwia usunięcie nieużytków i uporządkowanie pamięci operacyjnej. Wywołuje się ją przez jednoczesne naciśnięcie klawiszy: ALT i F10. Działanie procedury porządkowania pamięci jest sygnalizowane w sposób opisany w rozdziałach 3 i 8.

Opcja "Opcje"

Opcję "Opcje" wybiera się naciskając klawisz F6. Po jej uaktywnieniu na ekranie pojawia się następujące pod-menu:



Opcja "Autotranslacja"

Opcja ta umożliwia zmianę zachowania się edytora ekranowego po naciśnięciu klawisza ESC. Jeśli wartością opcji jest "Tak", to naciśnięcie klawisza ESC powoduje automatyczne zdefiniowanie procedur zawartych w edytorze ekranowym. Jeśli wartością opcji jest "Nie", to po naciśnięciu klawisza ESC system AC-LOGO zadaje pytanie:

Zdefiniować procedury?

W zależności od odpowiedzi procedury zawarte w edytorze ekranowym są definiowane lub nie.

Opcja "Praca krokowa"

Opcja ta umożliwia włączenie lub wyłączenie trybu, w którym wykonanie każdej procedury pierwotnej LOGO jest poprzedzone wyświetleniem komunikatu w dolnym wierszu ekranu i oczekiwaniem na wciśnięcie dowolnego klawisza.

Opcja "Śledzenie"

Opcja ta umożliwia włączanie i wyłączanie śledzenia wykonywanych procedur. Stan przełącznika śledzenia (Tak/Nie) jest widoczny obok nazwy opcji. Jego zmianę uzyskuje się przez naciśnięcie klawisza ENTER. Ustawienie przełącznika w pozycję "Tak", powoduje pojawienie się na ekranie okienka śledzenia. Okienko to jest podzielone na dwie części. Górna część przeznaczona jest do wyświetlania tekstów śledzonych procedur, a dolna część do wyświetlania śladów wywołań śledzonych procedur. Położenie i rozmiary okienka śledzenia można ustalić korzystając z opcji "Okna", lub klawiszy pomocniczych Ctrl-F2, Ctrl-F3, bądź też za pomocą myszy — ciągnąc ramkę okna śledzenia przy naciśniętym lewym przycisku myszy.

Sposób włączania śledzenia procedur LOGO opisano w rozdziale 8 procedury WŁĄCZŚLAD i WYŁĄCZŚLAD (ang. TRACE i UNTRACE).

Opcja "Opóźnienie interpretacji"

Opcja ta umożliwia zmianę szybkości interpretacji. Po jej wybraniu, na ekranie pojawia się okienko z uaktywnionym edytorem wierszowym i z wypisanym opóźnieniem (w milisekundach) realizowania każdej procedury pierwotnej języka LOGO. Opóźnienie to można zmieniać od zera (szybkość maksymalna) do 65534 ms. Wpisanie wartości 65535 powoduje uruchomienie trybu pracy krokowej, w którym wykonanie każdej procedury pierwotnej LOGO jest poprzedzone wyświetleniem specjalnego okienka i oczekiwaniem na wciśnięcie dowolnego klawisza.

Opcja "Rozmiar tabulacji"

Opcja ta umożliwia zmianę rozmiaru tabulacji w tekście zawartym w edytorze ekranowym. Bezpośrednio po wybraniu tej opcji, na ekranie pojawi się okienko z uaktywnionym edytorem wierszowym i z wpisaną dotychczasową wartością rozmiaru tabulacji. Jeśli po wprowadzeniu nowego rozmiaru tabulacji, w edytorze ekranowym znajduje się jakiś tekst, to nie będzie on automatycznie przeformatowywany.

Opcja "Zachowaj opcje"

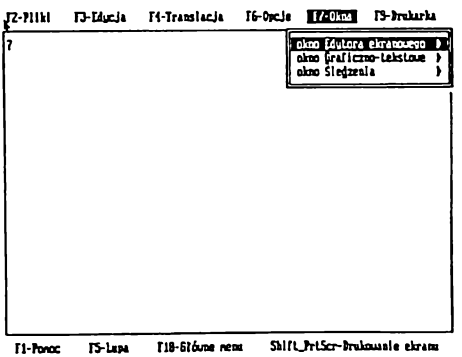
Opcja ta umożliwia zapisanie wszystkich parametrów systemu LOGO w pliku konfiguracyjnym o nazwie LOGO.CFG w katalogu, z którego uruchomiono system programowania AC-LOGO. Plik LOGO.CFG jest automatycznie wczytywany podczas ładowania systemu AC-LOGO. Sposób wykorzystania poleceń z pliku konfiguracyjnego opisano w dodatku F.

Opcja "Wczytaj opcje"

Opcja ta umożliwia wczytanie poleceń dla systemu LOGO z pliku konfiguracyjnego o nazwie LOGO.CFG, znajdującego się w katalogu, z którego uruchomiono system programowania AC-LOGO. Istnieje możliwość wykorzystania wszystkich poleceń z tego pliku poza deklarowaniem i wczytywaniem nowych generatorów znaków — polecenia z pliku konfiguracyjnego dotyczące definiowania nowych generatorów znaków są ignorowane. Opis poleceń z pliku konfiguracyjnego zawarto w dodatku F.

Opcja "Okna"

Opcję "Okna" wybiera się przez naciśnięcie klawisza F7. Po jej uaktywnieniu na ekranie pojawia się następujące pod-menu:



Opcja "okno Edytora ekranowego"

Opcja ta pozwala użytkownikowi na ustalenie wszystkich parametrów opisujących okienko edytora ekranowego, zgodnie z jego życzeniami. Po jej wybraniu, na ekranie pojawiają się kolejne zestawy opcji.

Pod-opcja "Zmiana rozmiaru i położenia"

Opcja ta umożliwia zmianę zarówno rozmiaru, jak i położenia okienka edytora ekranowego. Jeżeli okno edytora jest niewidoczne, to na czas zmiany rozmiaru i położenia pojawia się na ekranie, a po zakończeniu tej operacji znika.

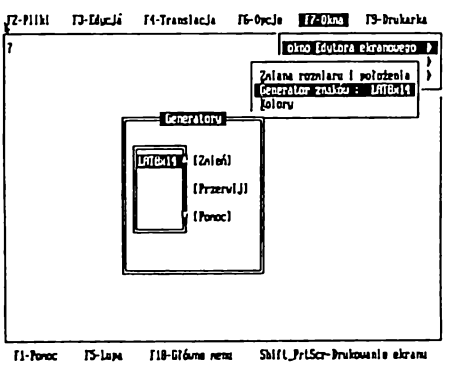
Przy pomocy klawiatury położenie okienka można zmieniać klawiszami "←", "→", "↑", "↓". Rozmiary okienka (tj. położenie dolnej i prawej krawędzi) zmienia się przy pomocy tych samych klawiszy połączonych z klawiszem SHIFT. Po zakończeniu modyfikacji należy nacisnąć klawisz ENTER.

Zmianę położenia okienka przy pomocy myszki realizuje się przez wskazanie dowolnego boku okienka, naciśnięcie lewego przycisku myszy, przesunięcie okienka (przy wciśniętym przycisku) na żądane miejsce i zwolnienie przycisku. Zmianę rozmiaru okienka realizuje się przez wskazanie dowolnego narożnika okienka, i powtórzenie opisanych wyżej czynności.

Bezpośrednio po wybraniu opcji zmiany rozmiaru i położenia, zawartość okienka edytora jest usuwana z ekranu. Po zakończeniu modyfikacji, zawartość jest przywracana.

Pod-opcja "Generator znaków"

Opcja ta pozwala użytkownikowi na ustalenie generatora znaków. Sposób tworzenia plików zawierających własne generatory znaków użytkownika opisano w dodatku G. Po wybraniu tej opcji, na ekranie pojawia się następujące okienko:



Wewnątrz znajduje się mniejsze okienko zawierające ośmioliterowe nazwy dostępnych generatorów znaków. Wyboru generatora dokonujemy za po-

mocą klawiszy przesuwu kursora w górę lub w dół. Wybór potwierdzamy naciśnięciem klawisza ENTER. W celu wybrania generatora za pomocą myszy, należy wskazać generator kursorem myszy i dwukrotnie nacisnąć jej lewy przycisk.

Pod-opcja "Kolory"

Opcja ta pozwala użytkownikowi na określenie wszystkich kolorów, związanych z okienkiem edytora ekranowego. Po wybraniu tej opcji, zależnie od posiadanej karty graficznej, na ekranie pojawia się okienko z opisem dostępnych kolorów.

Ramka "Parametry" zawiera wszystkie parametry opisujące kolory występujące w okienku edytora ekranowego. Ramka "Kolory" zawiera wszystkie kolory jakie są dostępne przy użyciu posiadanej karty graficznej. Ramka "Nagłówek" służy jako wzorzec przy ustalaniu kolorów poszczególnych parametrów. Obok nazw parametrów i kolorów są wyświetlone kolorowe znaczniki, ułatwiające użytkownikowi podjęcie decyzji.

Bezpośrednio po pojawieniu się okienka na ekranie, aktywna jest ramka "Parametry". Podświetlenie dowolnej nazwy parametru powoduje równoczesne podświetlenie, w ramce "Kolory", nazwy koloru związanego z tym parametrem. Zmianę koloru żądanego parametru realizuje się przez podświetlenie jego nazwy, naciśnięcie klawisza ENTER, co spowoduje uaktywnienie ramki "Kolory", podświetlenie wybranej nazwy koloru i ponowne naciśnięcie klawisza ENTER. Po dokonaniu tych operacji, ramką aktywną staje się znowu ramka "Parametry". W celu poinformowania systemu LOGO o dokonanych zmianach, należy wykonać operację "[Zmień kolory]". W tym celu należy nacisnąć klawisz "Tab", co spowoduje uaktywnienie napisu u dołu okienka, a następnie nacisnąć klawisz ENTER. Użycie klawisza ESC, lub któregoś z klawiszy pomocniczych, przy aktywnej opcji zmiany kolorów, powoduje jej przerwanie, a tym samym pozostawienie dotychczasowych kolorów.

Ustalenie kolorów przy pomocy myszki sprowadza się do wskazania najpierw nazwy parametru a potem nazwy żądanego koloru. W celu zatwierdzenia dokonanych zmian należy wskazać dowolny znak wchodzący w skład tekstu u dołu okienka i nacisnąć lewy przycisk myszki.

Opcja "okno Graficzno-tekstowe"

Opcja ta pozwala użytkownikowi na ustalenie parametrów opisujących okienko graficzno-tekstowe. Po jej wybraniu, na ekranie pojawiają się kolejne zestawy opcji.

Pod-opcja "Zmiana podziału ekranu"

Opcja ta umożliwia zmianę podziału ekranu między okienko graficz-

no-tekstowe i okienko śledzenia. Jeśli orientacja okienka śledzenia jest pozioma, wówczas, przy pomocy klawiszy: "↑" i "↓" można zmieniać wysokość obu okienek. Jeśli orientacja okienka śledzenia jest pionowa, wówczas, przy pomocy klawiszy: "←" i "→" można zmieniać ich szerokość. Zatwierdzenia dokonanego podziału dokonuje się przez naciśnięcie klawisza ENTER.

Zmianę podziału ekranu za pomocą myszki realizuje się przez wskazanie linii dzielącej okienka: graficzno-tekstowe i śledzenia, naciśnięcie lewego przycisku myszki, przesunięcie (przy wciśniętym przycisku) wskazanej linii na żądaną pozycję i zwolnienie przycisku.

Uwaga !!!

Bezpośrednio po wywołaniu niniejszej opcji zawartości okienek: graficzno-tekstowego i śledzenia są bezpowrotnie tracone.

Pod-opcja "Generator znaków"

Opcja ta pozwala użytkownikowi na ustalenie generatora znaków dla okienka graficzno-tekstowego. Sposób wyboru generatora znaków dla tego okienka jest identyczny ze sposobem opisanym dla opcji "Generator znaków dla okna edytora".

Pod-opcja "Kolory"

Opcja ta pozwala użytkownikowi na ustalenie wszystkich kolorów związanych z oknem graficzno-tekstowym. Sposób ustalania kolorów jest identyczny ze sposobem opisanym dla opcji "Kolory dla okna edytora".

Opcja "okno Śledzenia"

Opcja ta pozwala użytkownikowi na ustalenie wszystkich parametrów opisujących okienko śledzenia. Po jej wybraniu, na ekranie pojawiają się kolejne zestawy opcji.

Pod-opcja "Orientacja"

Opcja ta umożliwia zmianę orientacji okienka śledzenia. Orientacja pozioma oznacza, że okienko śledzenia będzie umieszczone na dole ekranu i będzie miało szerokość równą szerokości okienka graficzno-tekstowego. Orientacja pionowa oznacza, że okienko śledzenia będzie umieszczone z prawej strony ekranu i będzie miało wysokość równą wysokości okienka graficzno-tekstowego. Wyboru orientacji (po uprzednim podświetleniu opcji "Orientacja") dokonuje się na zasadzie przełącznika, tzn. każdorazowe

naciśnięcie klawisza ENTER powoduje zmianę orientacji na przeciwną. Aktualna orientacja jest wyświetlona obok nazwy opcji.

Uwaga III:

Po wybraniu niniejszej opcji zawartości okienek: graficzno-tekstowego i śledzenia są bezpowrotnie tracone.

Pod-opcja "Zmiana podziału okienka"

Opcja ta umożliwia użytkownikowi zdefiniowanie własnego podziału okienka śledzenia. Realizuje się to przez przesunięcie linii oddzielającej okienko z tekstami śledzonych procedur od okienka ze śladami wywołań. W zależności od orientacji okienka śledzenia, linię oddzielającą przesuwa się albo w kierunku pionowym, albo w kierunku poziomym.

Przy pomocy klawiatury przesunięcia dokonuje się przy użyciu klawiszy: "←" i "→" (przy orientacji poziomej), lub "↑" i "↓" (przy orientacji pionowej). W celu zatwierdzenia dokonanego podziału należy nacisnąć klawisz ENTER.

Przy pomocy myszki przesunięcia dokonuje się przez wskazanie linii dzielącej okienka, naciśnięcie lewego przycisku, i przesunięcie (przy wciśniętym przycisku) wskazanej linii na żądaną pozycję.

Uwaga III:

Wybranie niniejszej opcji powoduje, że zawartość okienka śledzenia jest bezpowrotnie tracona.

Pod-opcja "Generator znaków"

Opcja ta pozwala użytkownikowi na ustalenie generatora znaków dla okienka śledzenia. Sposób wyboru generatora znaków dla tego okienka jest identyczny ze sposobem opisanym dla opcji "Generator znaków dla okna edytora".

Pod-opcja "Kolory"

Opcja ta pozwala użytkownikowi na ustalenie wszystkich kolorów związanych z oknem śledzenia. Sposób ustalania kolorów jest identyczny ze sposobem opisanym dla opcji "Kolory dla okna edytora".

Opcja "Drukarka"

Opcję "Drukarka" wybiera się przez naciśnięcie klawisza F9. Po jej uaktywnieniu na ekranie pojawiają się kolejne zestawy opcji.

Dokładny opis parametrów wydruku kopii ekranu zawarto w rozdziale 8, w części dotyczącej procedur pierwotnych MARGINESYDRUKU, TRYBDRUKU i SKALUJDRUK (ang. SETCOPYMARGIN, SETCOPYMODE, SETCOPYSCALE).

Pod-opcja "Zerowanie drukarki"

Opcja ta powoduje wyzerowanie drukarki (bez konieczności ponownego ładowania wzorów znaków narodowych).

Pod-opcja "Przejście do nowej strony"

Opcja ta powoduje wysłanie do drukarki komendy wysuwu papieru do nowej strony.

Pod-opcja "Kopia ekranu monitora"

Opcja ta pozwala użytkownikowi na określenie parametrów wydruku zawartości ekranu monitora. Po jej wybraniu na ekranie pojawia się kolejny zestaw opcji.

Pod-opcja "Powiększenie"

Opcja ta umożliwia ustalenie powiększenia wydruku kopii ekranu. Po jej wybraniu na ekranie pojawia się okienko składające się z następujących elementów:

- pola "Powiększenie poziome", w którym jest wyświetlane aktualne powiększenie wydruku w poziomie,
- pola "Powiększenie pionowe", w którym jest wyświetlane aktualne powiększenie wydruku w pionie,
- pola "Ustal", którego wybranie powoduje przyjęcie wyświetlanego w okienku powiększenia w poziomie i w pionie,
- pola "Zaniechaj", którego wybranie powoduje powrót do menu bez zmiany powiększenia,
- pola "Pomoc", którego wybranie powoduje wywołanie systemu pomocy.

W celu zmiany powiększenia należy za pomocą klawiszy TAB, SHIFT-TAB, "↑", "↓" lub myszy przejść do pola opisującego powiększenie poziome lub pionowe. Następnie należy wybrać jedno z dostępnych powiększeń posługując się klawiszami "←" , "→", klawiszem ENTER lub lewym klawiszem myszy. Po wybraniu powiększenia należy za pomocą klawiszy TAB, SHIFT-TAB, "↑", "↓" lub myszy przejść do pola "Ustal" i nacisnąć ENTER.

Pod-opcja "Margines"

Opcja ta umożliwia ustalenie lewego i górnego marginesu dla wydruku. Po jej wybraniu na ekranie pojawia się okienko składające się z następujących elementów:

- pola "Margines lewy", w którym jest wyświetlany aktualny rozmiar marginesu z lewej strony wydruku (w znakach),
- pola "Margines górny", w którym jest wyświetlany aktualny rozmiar górnego marginesu wydruku (w znakach),
- pola "Ustal", którego wybranie powoduje przyjęcie wyświetlanych w okienku marginesów,
- pola "Zaniechaj", którego wybranie powoduje powrót do menu bez zmiany marginesów,
- pola "Pomoc", którego wybranie powoduje wywołanie systemu pomocy.

Marginesy ustala się w sposób podobny do ustalania powiększenia. Różnica polega na tym, że w polach "Margines lewy" i "Margines górny" nie wybiera się spośród dostępnych wartości, ale bezpośrednio wpisuje się żądany rozmiar.

Pod-opcja "Orientacja wydruku"

Opcja ta pozwala na określenie orientacji wydruku. Działa na zasadzie przełącznika. Aktualny stan przełącznika (Pozioma/Pionowa) jest wyświetlony obok nazwy opcji.

Pod-opcja "Tryb graficzny"

Opcja ta pozwala na określenie trybu graficznego drukarki. Po jej wybraniu, na ekranie pojawia się okienko wyboru trybu. W lewej, górnej części okienka są wyświetlane rodzaje drukarek, w prawej — okienko do wprowadzania trybu pracy drukarki, a poniżej — pola: "Ustal", "Zaniechaj", "Pomoc", których znaczenie jest takie samo jak w wyżej opisanych opcjach. Zasady ustalania trybu graficznego są podobne do zasad ustalania powiększenia.

Pod-opcja "Jakość druku"

Opcja ta pozwala na określenie jakości druku. Działa na zasadzie przełącznika. Aktualny stan przełącznika (druk niskiej jakości "Draft"/ druk wysokiej jakości "NLQ") wyświetlony jest obok nazwy opcji. Jakość druku dotyczy zarówno wydruku tekstu edytora, jak i kopii ekranu.

Pod-opcja "Drukuj..."

Opcja ta powoduje wydrukowanie zawartości ekranu monitora na drukarce zgodnie z ustalonymi parametrami. Podobny efekt można uzyskać po naciśnięciu kombinacji klawiszy Shift i PrtScr (Print Screen).

Pod-opcja "Wydruk tekstu edytora"

Opcja ta powoduje wydrukowanie zawartości obszaru roboczego edytora na drukarce zgodnie z ustalonymi parametrami.

Pod-opcja "Margines"
Pod-opcja "Jakość druku tekstu"

Opcje te są identyczne z opisanymi wcześniej opcjami dla wydruku kopii ekranu.

Pod-opcja "Wydruk z przerwą po stronie"

Opcja ta pozwala na ustalenie wydruku tekstu zawartego w edytorze ekranowym jako ciągłego, lub strona po stronie (z przerwą po każdej wydrukowanej stronie). Działa na zasadzie przełącznika o stanach: Tak/Nie.

Pod-opcja "Zdefiniowanie znaków narodowych"

Opcja ta powoduje przesłanie do drukarki wzorów polskich znaków narodowych oraz wzorów znaków graficznych, odczytanych z pliku LDRAFT.PFT lub LNLQ.PFT (w zależności od ustawienia opcji Draft/NLQ opisanej wcześniej). Nazwy plików zawierających wzory znaków można zmieniać przy pomocy poleceń pliku konfiguracyjnego.

Pod-opcja "Drukuj..."

Opcja ta powoduje wydrukowanie tekstu zawartego w edytorze ekranowym zgodnie z ustalonymi parametrami.

7.3. System podpowiedzi

System podpowiedzi jest interakcyjnym systemem wyświetlania tekstów i rysunków zawierających informacje na temat systemu programowania AC-LOGO. Informacje te są ze sobą powiązane. Wyświetlany tekst zawiera wyróżnione słowa — hasła. Dla każdego z haseł jest pamiętany jeden tekst podpowiedzi lub jeden rysunek. Teksty oraz rysunki można kolejno wyświet-

lać na ekranie, wskazując wybrane hasło. Wszystkie hasła są zgrupowane w indeks.

System podpowiedzi może być uaktywniony na cztery sposoby. Pierwszy z nich polega na wciśnięciu klawisza F1. Na ekranie jest wyświetlane okienko z hasłem opisującym aktualny kontekst systemu AC-LOGO. Powtórne naciśnięcie klawisza F1 powoduje wyświetlenie zestawu najważniejszych haseł. Wybór hasła następuje po przesunięciu kursora tekstowego nad wybrane hasło i naciśnięciu klawisza ENTER. Przesuwanie kursora można wykonywać za pomocą następujących klawiszy:

strzałki — o jeden znak,

PgUp, PgDn — o stronę,

Home, End — na początek/koniec wiersza,

Tab, Shift-Tab — na następne/poprzednie hasło.

W celu wyboru hasła za pomocą myszy należy przesunąć jej kursor nad wybrane hasło i dwukrotnie nacisnąć jej lewy przycisk. Wybór hasła powoduje wyświetlenie tekstu podpowiedzi dotyczącego tego hasła. Wyświetlony tekst może zawierać inne hasła, które można wybierać itd. Po odszukaniu właściwego hasła i przeczytaniu tekstu podpowiedzi można usunąć okienko podpowiedzi z ekranu — należy w tym celu nacisnąć klawisz ESC lub prawy przycisk myszki. Można także powrócić do wyświetlania poprzedniego okienka podpowiedzi, po naciśnięciu kombinacji klawiszy Alt-F1.

Drugi sposób uaktywnienia systemu podpowiedzi polega na naciśnięciu kombinacji klawiszy Ctrl-F1. W takim wypadku system stara się odszukać tekst podpowiedzi dla hasła, na które wskazuje kursor tekstowy na ekranie. Ten sposób jest pomocny w przypadku pracy w edytorze ekranowym lub wierszowym, gdy chcemy szybko poznać informacje na temat wybranej procedury pierwotnej. Wskazujemy kursorem nazwę tej procedury i naciskamy klawisze Ctrl-F1. Na ekranie zostanie wyświetlone okienko z tekstem podpowiedzi odnoszącym się do wskazanej procedury. Jeżeli system podpowiedzi nie zawiera informacji na temat wskazanego hasła, to jest wyświetlany indeks wszystkich haseł z wyróżnionym tym hasłem, które jest najbardziej zbliżone do wskazanego. Użytkownik może w takim wypadku wcisnąć ENTER i wyświetlić informacje na temat hasła zbliżonego do wskazanego, lub wybrać inne pokrewne hasło.

Trzeci sposób wywołania systemu podpowiedzi polega na naciśnięciu jednocześnie klawiszy ALT i F1. Na ekranie zostanie wyświetlone ostatnio wywołane okienko podpowiedzi. Naciskając ponownie kombinację klawiszy ALT-F1 można sukcesywnie przywoływać na ekran poprzednie okienka podpowiedzi, w odwrotnej kolejności niż były one wyświetlane po raz pierwszy. W systemie są pamiętane informacje na temat 16 ostatnio wyświetlanych okienek podpowiedzi.

Czwarty, ostatni sposób wywołania systemu podpowiedzi polega na naciśnięciu klawiszy SHIFT i F1. Na ekranie zostanie wyświetlony indeks wszystkich haseł podpowiedzi, uporządkowany alfabetycznie. Po wyświetleniu indeksu można odszukać dane hasło i nacisnąć klawisz ENTER, wyświetlając okienko podpowiedzi na temat tego hasła. W każdej chwili można powrócić do wyświetlania indeksu haseł przez naciśnięcie klawiszy SHIFT i F1.

7.4. Opis edytora wierszowego dla systemu menu

Edytor wierszowy dla systemu menu umożliwia wprowadzenie jednego wiersza tekstu o maksymalnej długości 255 znaków. Jeśli wprowadzany tekst jest dłuższy niż szerokość okienka z uaktywnionym edytorem wierszowym, następuje przewijanie poziome tekstu, co jest sygnalizowane przez pojawienie się znaków "«" lub "»" na krańcach okienka. Pełny opis edytora zawarto w rozdziale 5, dlatego tutaj ograniczymy się tylko do wymienienia funkcji klawiszy specjalnych i sterujących.

Klawisze sterujące:

"↑", "↓"	— dostęp do poprzednich zawartości edytora
"←", "→"	— przesunięcie kursora o jeden znak w lewo lub w prawo,
HOME, END	— ustawienie kursora na początku lub na końcu wprowadzanego tekstu,
BS	— zmazanie znaku przed kursorem,
DEL	— zmazanie znaku pod kursorem,
CTRL-Y	— zmazanie całego wiersza,
INS	— zmiana trybu wpisywania tekstu (wstawianie / zamiana),
TAB	— zaniechanie redagowania tekstu,
Esc	— zaniechanie redagowania tekstu,
ENTER	— zakończenie wprowadzania tekstu.

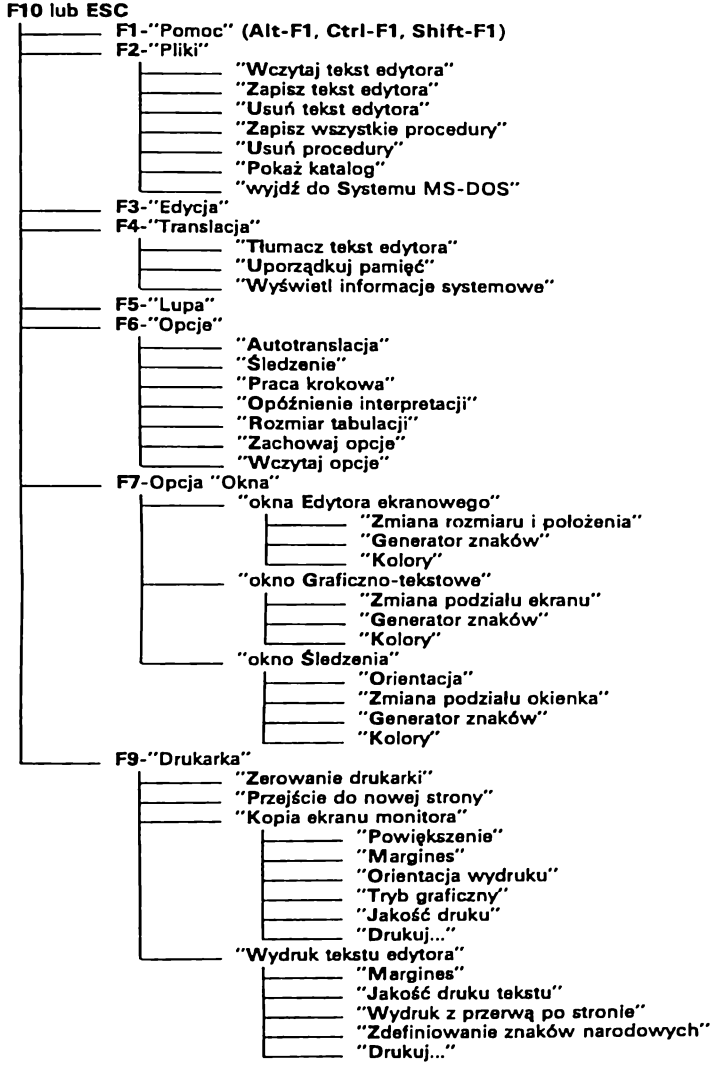
7.5. Opis klawiszy specjalnych

W systemie AC-LOGO zdefiniowano szereg klawiszy specjalnych, które mają za zadanie umożliwienie szybkiego wywoływania niektórych opcji systemu okienkowego. Naciśnięcie klawisza specjalnego (ewentualnie w połączeniu z klawiszem ALT, CTRL lub SHIFT) umożliwia przejście do danej opcji systemu okienkowego bez konieczności czasochłonnego wywoływania kolejnych okienek, prowadzących do okienka z szukaną opcją.

Klawisze specjalne nie są aktywne w trakcie wykonywania procedur LOGO — ich naciśnięcie podczas wykonywania programu nie pociąga za sobą żadnych skutków. Do klawiszy specjalnych zaliczamy następujące klawisze (kombinacje klawiszy).

F1	— wywołanie systemu podpowiedzi
ALT F1	— ponowne wyświetlenie poprzednio wyświetlanego okienka systemu podpowiedzi
CTRL-F1	— kontekstowe wywołanie systemu podpowiedzi
SHIFT F1	— wywołanie indeksu nazw systemu podpowiedzi
F2	— pliki
ALT-F2	— zapisanie procedur z obszaru roboczego edytora ekranowego na dysk
CTRL-F2	— zmiana podziału okna graficzno-tekstowego
F3	— wywołanie edytora ekranowego
ALT-F3	— wczytanie z pliku dyskowego tekstu do obszaru roboczego edytora ekranowego
CTRL-F3	— zmiana podziału okna śledzenia
F4	— wywołanie okienka "Translacja"
ALT-F4	— włączenie trybu śledzenia
CTRL-F4	— zmiana rozmiaru okna edytora ekranowego
F5	— wywołanie opcji "Lupa" czyli trybu pracy pełnoekranowej
ALT-F5	— maksymalizacja/minimalizacja rozmiaru okna edytora ekranowego
F6	— wywołanie okienka "Opcje"
F7	— wywołanie okienka "Okna"
F9	— wywołanie okienka "Drukarka"
ALT-F9	— przetłumaczenie procedur z obszaru roboczego edytora ekranowego (kompilacja)
F10	— główne menu
ALT-F10	— uporządkowanie pamięci obiektów LOGO (zlikwidowanie nieużytków z pamięci operacyjnej, odzyskanie miejsca w pamięci)
SHIFT-F10	— wyświetlenię wizytówki systemu AC-LOGO
SHIFT-PrtScr	— wydrukowanie zawartości ekranu na dołączonej drukarce

Mapka poruszania się po okienkowym systemie menu



LEKSYKON PROCEDUR PIERWOTNYCH

8. Leksykon procedur pierwotnych

Rozdział ten zawiera leksykon procedur pierwotnych języka AC–LOGO. Procedury są opisane w kolejności alfabetycznej, w następującym układzie:

NAZWA (skrót)	NAZWA ANGIELSKA (skrót)
Składnia:	składnia wywołania–nazwa par1 par2 ...→ wynik
Opis działania:	szczegółowy opis działania procedury
Parametry:	opis parametrów
Wynik:	opis wyniku (w przypadku funkcji)
Uwagi:	ewentualne uwagi i wskazówki
Przykłady:	przykładowy program

W opisie składni wywołania procedury *par1 par2* ... precyzuje liczbę, kolejność i typy parametrów. Zapis "→ *wynik*" nie jest elementem składni procedury, występuje on w przypadku procedur będących funkcjami w celu sprecyzowania typu wyniku. Typy wartości parametrów i wyniku są oznaczane w następujący sposób:

s — dowolne słowo;
n — liczba;
b — wartość logiczna;
l — lista.

Jeśli parametr lub wynik może przyjmować wartości dwóch typów, to są wymieniane oba dozwolone typy: na przykład

sl

oznacza, że parametr lub wynik może być słowem lub listą; na przykład w procedurze

BEZPIERW sl → sl

zarówno parametr, jak i wynik mogą być słowem lub listą.

Przykłady w tekście są opracowane w polskiej wersji języka LOGO. Wersję angielską otrzymuje się przez zastąpienie nazw procedur pierwotnych odpowiednimi nazwami angielskimi.

ABS **ABS**

Składnia: ABS n → n

Opis działania: funkcja, której wartością jest wartość bezwzględna parametru.

Parametr: wyrażenie arytmetyczne.

Wynik: liczba nieujemna.

Przykład

```
?pisz abs -3
3
?pisz abs 0
0
```

ARCCOS **ARCCOS**

Składnia: ARCCOS n → n

Opis działania: funkcja trygonometryczna, której wartością jest funkcja arcus cosinus parametru.

Parametr: wyrażenie arytmetyczne o wartości z zakresu (–1, 1).

Wynik: kąt w stopniach.

Przykłady

```
?pisz arccos 1
0
?pisz arccos 0
90
```

ARCSIN **ARCSIN**

Składnia: ARCSIN n → n

Opis działania: funkcja trygonometryczna, której wartością jest funkcja arcus sinus parametru.

Parametr: wyrażenie arytmetyczne o wartości z zakresu (–1, 1).

Wynik: kąt w stopniach.

Przykłady

```
?pisz arcsin 1
90
?pisz arcsin 0
0
```

ARCTG **ARCTAN**

Składnia: ARCTG n → n

Opis działania: funkcja trygonometryczna, której wartością jest funkcja arcus tangens parametru.

Parametr: wyrażenie arytmetyczne.

Wynik: kąt w stopniach.

Przykłady

```
?pisz arctg 1
45
?pisz arctg 0
0
```

ASCII **ASCII**

Składnia: ASCII sl → n

Opis działania: funkcja, której wartością jest rozszerzony kod ASCII pierwszego znaku obiektu będącego jej parametrem.

Parametr: wyrażenie, którego wartością jest słowo lub lista, której pierwszym elementem jest słowo.

Wynik: kod ASCII.

Przykłady

```
?pisz ascii "Ala
65
?pisz ascii ["Ala]
65
?pisz ascii [Ala]
65
?pisz ascii "123
49
?pisz ascii 123
49
?pisz ascii ["123]
49
?pisz ascii [123]
49
```

AZYMUT — TOWARDS

Składnia: AZYMUT l → n

Opis działania: funkcja, której wartością jest kąt mierzony zgodnie z kierunkiem ruchu wskazówek zegara, między pionem a prostą przechodzącą przez aktualną pozycję żółwia i punkt o współrzędnych podanych jako parametr.

Parametr: wyrażenie, którego wartością jest dwuelementowa lista postaci [n n] zawierająca współrzędne x i y punktu.

Wynik: kąt w stopniach.

Uwaga: jeżeli parametrem jest lista zawierająca aktualne współrzędne żółwia, to wartością funkcji jest zero.

Przykład

```
?wróć
?pisz azymut [10 10]
45
?pisz azymut [0 0]
0
```

BEZOST (BO) — BUTLAST (BL)

Składnia: BEZOST sl → sl

Opis działania: funkcja "bez ostatniego"; jej wartością jest obiekt będący parametrem, pozbawiony ostatniego elementu. Jeśli parametr jest obiektem pustym, to wartością funkcji jest obiekt pusty tego samego typu co parametr.

Parametr: wyrażenie, którego wartością jest słowo lub lista.

Wynik: jeśli parametrem jest słowo, to wynikiem jest słowo bez ostatniego znaku, a jeśli parametrem jest lista, to wynikiem jest lista bez ostatniego elementu.

Uwaga: podlisty są traktowane jako pojedyncze elementy listy.

Przykłady

```
?pokaż bezost [[Ala Ola] Jan]
[ [ Ala Ola ] ]
?pokaż bezost [Ala [Ola Jan]]
[ Ala ]
?pokaż bezost bezost [[Ala Ola] Jan]
[ ]
```

```
?pisz bezost "Ala
Al
?pokaż bezost []
[ ]
?pokaż bezost ''
''
?pisz bezost 123
12
?pisz bezost 1 > 0
PRAWD
```

BEZPIERW (BP) **BUTFIRST (BF)**

Składnia: BEZPIERW sl → sl

Opis działania: funkcja "bez pierwszego"; jej wartością jest obiekt będący parametrem, pozbawiony pierwszego elementu. Jeśli parametr jest obiektem pustym, to wartością funkcji jest obiekt pusty tego samego typu co parametr.

Parametr: wyrażenie, którego wartością jest słowo lub lista.

Wynik: jeśli parametrem jest słowo, to wynikiem jest słowo bez pierwszego znaku; jeśli parametrem jest lista, to wynikiem jest lista bez pierwszego elementu.

Uwaga: podlisty są traktowane jako pojedyncze elementy listy.

Przykłady

```
?pokaż bezpierw [[Ala Ola] Jan]
[ Jan ]
?pokaż bezpierw [Ala [Ola Jan]]
[ [ Ola Jan ] ]
?pokaż bezpierw bezpierw [Ala [Ola Jan]]
[ ]
?pisz bezpierw "Ala
la
?pokaż bezpierw []
[ ]
?pokaż bezpierw ''
''
?pisz bezpierw 123
23
?pisz bezpierw 1 = 0
AŁSZ
```

BO

Patrz BEZOST

BP

Patrz BEZPIERW

CL

Patrz CZYTAJLISTĘ

COS **COS**

Składnia: COS n → n

Opis działania: funkcja trygonometryczna cosinus.

Parametr: wyrażenie, którego wartością jest kąt w stopniach.

Wynik: liczba równa cosinusowi parametru.

Przykłady

```
?pisz cos 0
1
?pisz cos 60
0.5
```

CRS

Patrz CZYTAJRS232

CS

Patrz CZYŚĆ

CZ

Patrz CZYTAJZNAK

CZAS **TIME**

Składnia: CZAS → l

Opis działania: funkcja, ktorej wartością jest lista określająca czas systemowy w formie [godzina minuta sekunda setne_sekundy]

Przykłady

```
?ps czas
12 21 35 70
```

CZEKAJ **WAIT**

Składnia: CZEKAJ n

Opis działania: procedura zawieszająca działanie programu na okres określony przez parametr.

Parametr: wyrażenie arytmetyczne o wartości z zakresu od 0 do 65535 — czas zawieszenia w 1/50 s.

Przykłady

```
?grafika czekaj 100
{Na okres dwóch sekund cały ekran zostaje przeznaczony na grafikę}
?teksty
?oto syrenka :czas :przerwa
> powtórz :czas
>  [ nuta [2 2] czekaj :przerwa ]
>już
SYRENKA zdefiniowano
?SYRENKA 50 3
{Z głośnika dochodzi odgłos ruszającego samochodu}
```

CZYŚĆ (CS) **CLEARSCREEN (CS)**

Składnia: CZYŚĆ

Opis działania: procedura umieszczająca żółwia w początku układu współrzędnych, ustawiająca go pionowo w górę i wypełniająca okienko tekstowo-graficzne kolorem tła. Liczba wierszy tekstu w dolnej części ekranu jest określona przez parametr uprzednio wykonanej procedury DZIELEKRAN (ang. SETSPLIT) lub jest przyjmowana domyślnie (4) po wywołaniu systemu programowania AC–LOGO.

Uwaga: procedura zachowuje efekty uprzednio wykonanych procedur SŻ (ang. HIDETURTLE) i PŻ (ang. SHOWTURTLE).

Przykład

```
?sż
?czyść
```

{Żółw jest ustawiany na początku układu współrzędnych, pionowo w górę, lecz jest niewidoczny ze względu na uprzednie wykonanie procedury SŻ}

CZYTAJLISTĘ (CL) **READLIST (RL)**

Składnia: CZYTAJLISTĘ → l

Opis działania: funkcja, której wartością jest lista słów składających się na tekst wprowadzony z klawiatury. Wywołanie tej funkcji zawiesza działanie programu w oczekiwaniu na wprowadzenie z klawiatury jednego wiersza tekstu zakończonego znakiem nowego wiersza CR.

Wynik: lista słów wprowadzonych z klawiatury, bez znaku CR.

Przykład

```
?oto powitanie
> pisz [Jak masz na imię?]
> {wczytanie imienia w mianowniku}
> przypisz "imię pierw czytajlistę
> {tworzenie imienia w wołaczu}
> przypisz "końcówka słowo ost bezost :imię ost :imię
> wybór :końcówka
> [
>   "ia [przypisz "imię słowo bezost :imię "u]
>   "ek [przypisz "imię słowo bezost bezost :imię "ku]
>   "da [przypisz "imię słowo bezost :imię "o]
> { pozostałe reguły: ... }
>   "inaczej [przypisz "imię " ] {nieznana regula}
> ]
> (pisz [Cześć] :imię "!)
>już
POWITANIE zdefiniowano
?oto wybór :wzorzec :działanie
>{Realizacja odpowiednika instrukcji case w języku Pascal}
> jeśli puste? :działanie [stop]
> jeśli lub równe? :wzorzec pierw :działanie
>            równe? "inaczej pierw :działanie
> [zrób pierw bezpierw :działanie stop]
> wybór :wzorzec bezpierw :działanie
>już
```

```
WYBÓR zdefiniowano
?powitanie
Jak masz na imię?
Jarek
Cześć Jarkul
?powitanie
Jak masz na imię?
Magdunia
Cześć Magduniul
?powitanie
Jak masz na imię?
Krzyś
Cześć !
```

CZYTAJRS232 **READSERIALCHAR (RSC)**

Składnia: CZYTAJRS232 → s

Opis działania: wczytanie jednego znaku ze złącza transmisji szeregowej RS–232C.

Wynik: jednoznakowe słowo zawierające znak wczytany z RS–232C.

Uwaga: funkcja powoduje zawieszenie działania programu aż do czasu odebrania znaku z RS–232C.

Przykłady

```
?ps czytajrs2322
a
```

CZYTAJZNAK (CZ) **READCHAR (RC)**

Składnia: CZYTAJZNAK → s

Opis działania: funkcja, której wartością jest jednoznakowe słowo zawierające wczytany znak. Wywołanie tej funkcji zawiesza działanie programu w oczekiwaniu na wciśnięcie klawisza.

Wynik: słowo jednoznakowe.

Przykład

```
?oto armata :pozycja
> kursor zdanie 0 :pozycja
> wpisz "=–
> kursor [0 20]
> podnieś poz zdanie –121 92 – 8 * :pozycja
> kąt 90
```

```
) odwracanie
) jeśli nie równe? ml czytajznak "s
) {naciśnięcie klawisza "S" powoduje wystrzał z armaty,
)  innego klawisza — zakończenie działania programu}
)   [stop]
) naprzód 512
) armata :pozycja
)już
ARMATA zdefiniowano
?czyść sż armata 12
```

DATA **DATE**

Składnia: DATA → l

Opis działania: funkcja, ktorej wartością jest lista określająca datę systemową w formie [dzień miesiąc rok]

Przykłady

```
?ps data
26 03 1992
```

DŁUGOŚĆ **COUNT**

Składnia: DŁUGOŚĆ sl → n

Opis działania: funkcja, której wartością jest liczba elementów obiektu będącego parametrem.

Parametr: wyrażenie, którego wartością jest slowo lub lista.

Wynik: liczba znaków w slowie lub liczbą elementów listy.

Uwaga: podlisty są traktowane jako pojedyncze elementy listy.

Przykłady

```
?pisz dlugość "Magda
5
?pisz dlugość "'Magda jest ladna'
16
?pisz dlugość "
0
?pisz dlugość [Ala Ola Jan]
3
?pisz dlugość [[Ala Ola] Jan]
2
```

```
?pisz dlugość [[Ala Ola Jan]]
1?pisz dlugość []
0
?pisz dlugość 123
3
?pisz dlugość 1 > 0
6
?przypisz "lista zdanie "a '"b c d e' pisz :lista
a b c d e
?pisz dlugość :lista
2
?przypisz "lista zdanie " "
?pokaż :lista
[ ]
?pisz dlugość :lista
2
```

DOŚĆ **BYE**

Składnia: DOŚĆ

Opis działania: zakończenie pracy w LOGO i przejście do systemu operacyjnego MS-DOS. Kompilator upewnia się, czy nie nastąpiła pomyłka pisząc

'Jesteś pewien (t/n)?'

i żąda potwierdzenia decyzji klawiszem 'T'. Naciśnięcie dowolnego innego klawisza powoduje powrót do aktualnie wykonywanej procedury bez utraty tekstów procedur i zmiennych.

Przykład

```
?oto wyjście
> pisz [Czy chcesz zakończyć pracę?]
> dość
> pisz [Pracujemy dalej]
>już
WYJŚCIE zdefiniowano
?wyjście
Czy chcesz zakończyć pracę?
Jesteś pewien (t/n)?n
Pracujemy dalej
```

DRUKUJOBRAZ **COPYSCREEN (CPS)**

Składnia: DRUKUJOBRAZ

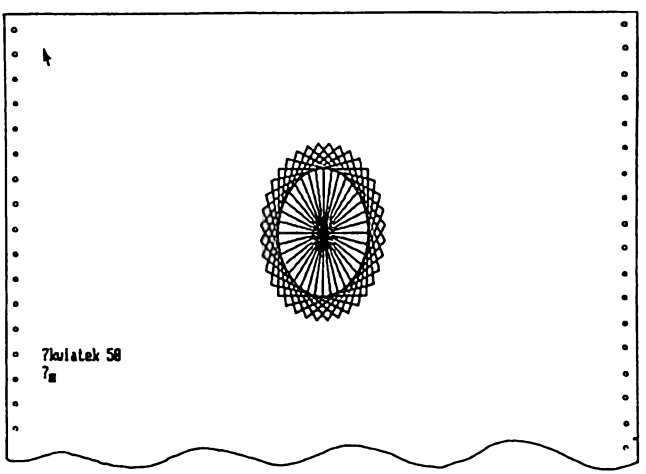
Opis działania: procedura powodująca skopiowanie na drukarce całego ekranu, łącznie z oknem tekstowym, ramką dookoła okienka

tekstowo-graficznego, menu itp. Rozmiar wydruku i jego położenie zależy od ostatnio wykonanych procedur SKALUJDRUK (ang. SETCOPYSCALE), ORIENTUJDRUK (ang. SETCOPYORIENTATION), MARGINESYDRUKU (ang. SETCOPYMARGIN) i TRYBDRUKU (ang. SETCOPYMODE).

Uwagi: jeśli drukarka nie jest podłączona, to kompilator zawiesza swą pracę oczekując na gotowość drukarki, wysyłając po kilkunastu sekundach oczekiwania odpowiedni komunikat na ekran. Sekwencje sterujące drukarkami można ustalać przy pomocy poleceń z pliku konfiguracyjnego (por. Dodatek F).

Przykład

```
?oto serweta :n
> powtórz :n [ romb 50 prawo 360 / :n ]
>już
SERWETA zdefiniowano
?oto romb :bok
> powtórz 2 [ naprzód :bok prawo 60 naprzód :bo prawo 120 ]
>już
ROMB zdefiniowano
?grafika sż
?serweta 60
?drukujobraz
```



Rys. 8.1. Wydruk ekranu uzyskany za pomocą procedury pierwotnej DRUKUJOBRAZ

DRUKUJOKNO **COPYWINDOW (CPW)**

Składnia: DRUKUJOKNO

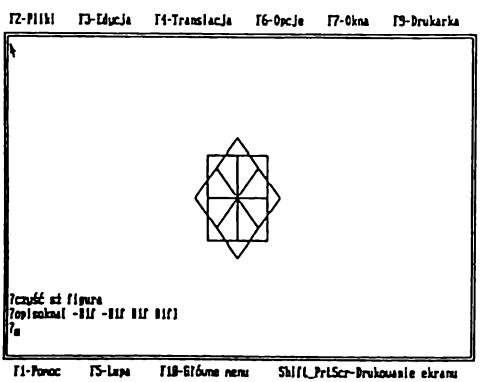
Opis działania: procedura powodująca skopiowanie na drukarce okienka graficznego z okna tekstowo-graficznego. Okienko tekstowe z dolnej części okienka nie jest kopiowane.

Uwaga: patrz DRUKUJOBRAZ

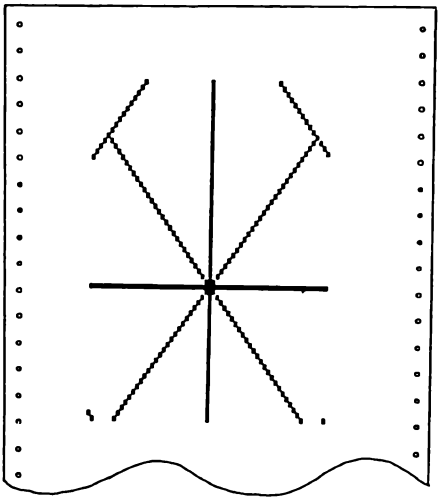
Przykład

```
?oto figura
> powtórz 8
> [
>   powtórz 4
>   [
>     naprzód 40
>     prawo 90
>   ]
>   prawo 45
> ]
>już
FIGURA zdefiniowano
?czyść sż figura
?opisokna [-#1F -#1F #1F #1F]
{Rozmiar okna graficznego zostaje zawężony do 64 * 64
punktów}
?drukujokno
{Na drukarce jest kopiowana tylko środkowa część rysunku}
```

a)



b)



Rys. 8.2. Przykład zastosowania procedury pierwotnej DRUKUJOKNO:
a) obraz ekranu; b) postać wydruku

DYSK **SETDRIVE, SETPATH**

Składnia: DYSK s

Opis działania: ustalenie dysku i katalogu bieżącego.

Parametr: wyrażenie, którego wartością jest słowo; jeśli drugim znakiem słowa jest dwukropek, to pierwszy znak jest traktowany jako nazwa dysku (z zakresu A do Z); pozostała część słowa jest traktowana jako nazwa katalogu aktualnego. Jeśli na końcu nazwy katalogu nie ma znaku ukośnika \, to wynikowa nazwa katalogu bieżącego jest uzupełniana o ten znak.

Uwaga: ze względu na specjalną funkcję ukośnika \ w LOGO należy w nazwach katalogów zawsze stosować dwuznak \ \.

Przykłady

```
?dysk "a:
?pisz dysk?
A
?dysk "A:\\LOGO
?pisz ścieżka?
\LOGO\
```

{Kopiowanie programu z jednej dyskietki na drugą, przy użyciu jednego napędu}

```
?usw
{Usunięcie z pamięci roboczej procedur i zmiennych}
?czytaj "domek
{Wczytanie programu z pliku domek.log }
{Wykonanie zamiany dyskietki w napędzie A:}
?zapisz "domek
{Zapis programu na nową dyskietkę}
```

DYSK **DRIVE**

Składnia: DYSK? → s

Opis działania: funkcja, której wartością jest nazwa dysku bieżącego.

Wynik: jednoznakowe słowo A, B, ..., Z.

Przykłady

```
?pisz dysk?
A
?dysk "b:
?pisz dysk?
B
```

DZIELEKRAN **SETSPLIT**

Składnia: DZIELEKRAN n

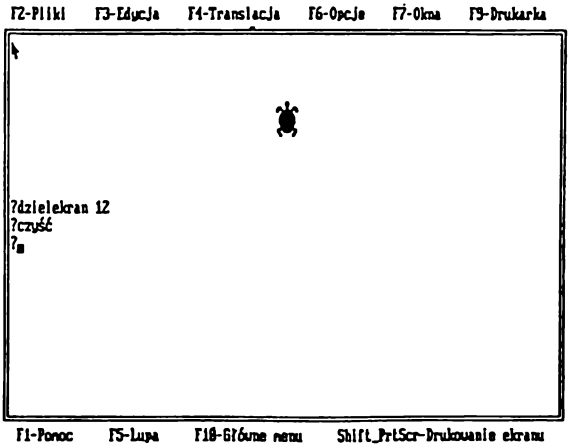
Opis działania: procedura ustalająca liczbę wierszy w oknie tekstowym.

Parametr: wyrażenie, którego wartością jest liczba wierszy w oknie tekstowym.

Uwagi: efekty wykonania procedury stają się widoczne dopiero po przejściu do graficzno-tekstowego trybu ekranu, np. po wykonaniu procedury CZYŚĆ (ang. CLEARSCREEN). Wielkość okna graficznego zależy od wybranego generatora znaków.

Przykład

```
?dzielekran 12
{Wygląd okienka tekstowo-graficznego nie ulega zmianie}
?czyść
{Górna połowa okienka tekstowo-graficznego zostaje przeznaczona na grafikę,
a dolna na teksty}
```



Rys. 8.3. Przykład zastosowania procedury pierwotnej DZIELEKRAN

ELEMENT — ITEM

Składnia: ELEMENT n sl → sl

Opis działania: funkcja, której wartością jest element obiektu podanego jako drugi parametr, o numerze określonym przez pierwszy parametr. Jeśli podany numer elementu jest większy niż liczba elementów obiektu lub jeśli ma wartość zero, to wartością funkcji jest obiekt pusty tego samego typu co parametr.

Parametry: pierwszym parametrem jest wyrażenie arytmetyczne, którego wartością jest nieujemna liczba określająca pozycję znaku w słowie lub elementu na liście; drugim parametrem jest wyrażenie, którego wartością jest słowo lub lista.

Wynik: jeśli drugi parametr jest słowem, to wartością funkcji jest jednoznakowe słowo lub słowo puste; jeśli drugi parametr jest listą, to wartością funkcji jest słowo lub lista.

Uwaga: podlisty są traktowane jako pojedyncze elementy.

Przykłady

```
?pisz element 1 [Ala [Ola Jan] Onufry]
Ala
?pokaż element 2 [Ala [Ola Jan] Onufry]
[ Ola Jan ]
```

```
?pisz element 1 element 2 [Ala [Ola Jan] Onufry]
Ola
?pokaż element 3 [Ala [Ola Jan] Onufry]
Onufry
?pokaż element 4 [Ala [Ola Jan] Onufry]
[ ]
?pokaż element 0 [Ala [Ola Jan] Onufry]
[ ]
?pisz element 4 "Onufry
f
?pisz element 7 "Onufry

?pisz element element 1 [3 2 1] [Ala Ola Jan]
Jan
?pisz element 2 2 > 0
R
```

ELEMENT? **MEMBERP**

Składnia: ELEMENT? sl l → b

Opis działania: funkcja, której wartością jest PRAWDA jeśli jej pierwszy parametr jest elementem listy podanej jako drugi parametr.

Parametry: pierwszym parametrem jest wyrażenie, którego wartością jest słowo lub podlista, będąca poszukiwanym elementem listy stanowiącej drugi parametr procedury.

Wynik: wartość logiczna.

Uwaga: podlisty są traktowane jako pojedyncze elementy listy.

Przykład

```
?pisz element? "Ala [Ala Ola Jan]
PRAWDA
?pisz element? "Jola [Ala Ola Jan]
FAŁSZ
?pisz element? "Ala [[Ala Ola] Jan]
FAŁSZ
?pisz element? "Ola [Ala Olaf Jan]
FAŁSZ
?pisz element? 123 [123 345 67]
PRAWDA
?pisz element? 123 [123.0 345.9 67.7]
PRAWDA
?pisz element? 123 ["123 "345 "67]
PRAWDA
```

```
?pisz element? "prawda [prawda falsz]
PRAWDA
?pisz element? 2 > 1 [PRAWDA FAŁSZ]
PRAWDA
?pisz element? 2 > 1 [2 > 1]
FAŁSZ
```

ENT

Patrz ENTIER

ENTIER **INT**

Składnia: ENTIER n → n

Opis działania: funkcja, której wartością jest największa liczba całkowita, nie większa od liczby podanej jako parametr.

Parametr: wyrażenie arytmetyczne.

Wynik: liczba całkowita.

Przykłady

```
?pisz ent 3.14
3
?pisz ent -3.14
-4
```

EXP **EXP**

Składnia: EXP n → n

Opis działania: funkcja eksponencjalna.

Parametr: wyrażenie arytmetyczne.

Wynik: liczba równa liczbie Eulera (podstawa logarytmów naturalnych) podniesionej do potęgi równej parametrowi.

Przykłady

```
?pisz exp 1
2.71828182845905
?pisz exp 1.5
4.48168907033806
```

```
?pisz ln exp 1.5
1.5
?pisz exp ln 1.5
1.5
```

FAŁSZ **FALSE**

Składnia: FAŁSZ → b

Opis działania: funkcja, której wartością jest wartość logiczna FAŁSZ.

Przykłady

```
?pisz równe? falsz "falsz
FAŁSZ
?pisz równe? falsz "FAŁSZ
PRAWDA
?pisz równe? falsz (2 + 2) = 5
PRAWDA
```

GR

Patrz GRAFIKA

GRAFIKA (GR) **FULLSCREEN (FS)**

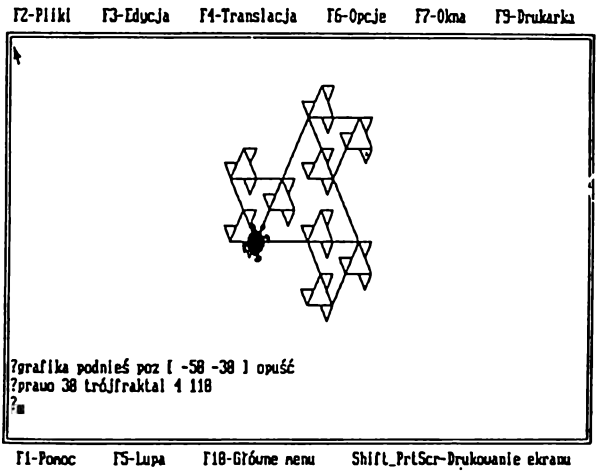
Składnia: GRAFIKA

Opis działania: procedura powodująca przeznaczenie całego okienka tekstowo-graficznego na okno graficzne. Procedura powoduje usunięcie tekstów oraz rysunków z okienka tekstowo-graficznego. Kursor tekstowy zostaje umieszczony w lewym, dolnym rogu okienka i nie jest wyświetlany do chwili wykonania procedury CZYTAJLISTĘ (ang. READLIST) lub CZYTAJZNAK (ang. READCHAR), albo do chwili przejścia do edytora wierszowego, w wyniku zakończenia wykonywania bieżącego programu.

Uwaga: po wykonaniu procedury GRAFIKA całe okienko graficzno-tekstowe jest przeznaczane na okno graficzne tylko na czas wykonywania procedur. Po przejściu do interpretacji poleceń użytkownika ostatni wiersz okienka jest przeznaczony na okno tekstowe, służące do ich wprowadzania.

Przykład

```
?oto trójfraktal :poziom :bok
> jeśli :poziom = 0 [stop]
> powtórz 3
> [
>   naprzód :bok
>   prawo 60
>   trójfraktal :poziom — 1 :bok / 2
>   prawo 60
> ]
>już
TRÓJFRAKTAL zdefiniowano
?grafika podnieś poz [—50 —30] opuść prawo 30
?trójfraktal 4 100
```



Rys. 8.4. Efekt wykonania procedury TRÓJFRAKTAL

I **AND**

Składnia: I b b → b
lub: (I b b ...) → b

Opis działania: funkcja, której wartością jest iloczyn logiczny parametrów.

Parametry: dwa wyrażenia o wartościach logicznych. W wersji zachłannej — dowolna, większa od jednego liczba wyrażeń o wartościach logicznych.

Wynik: iloczyn logiczny (koniunkcja) wartości logicznych parametrów.

Przykłady

```
?pisz i falsz falsz
FAŁSZ
?pisz i prawda falsz
FAŁSZ
?pisz i prawda prawda
PRAWDA
?pisz (i prawda prawda falsz)
FAŁSZ
?pisz (i 2 > 1 (abs -1) > 0 prawda)
PRAWDA
?przypisz "wiek 6
?jeśli i :wiek > 3 :wiek < 7 [pisz "przedszkolak]
przedszkolak
```

ILOCZYN **PRODUCT**

Składnia: ILOCZYN n n → n
lub: (ILOCZYN n n ...) → n

Opis działania: funkcja, której wartością jest iloczyn parametrów.

Parametry: dwa wyrażenia arytmetyczne. W wersji zachłannej — dowolna, większa od jednego liczba wyrażeń o wartościach liczbowych.

Wynik: liczba.

Przykłady

```
?pisz iloczyn 2 2
4
?pisz (iloczyn 2 3 4 -5)
-120
```

ILORAZC **QUOTIENT**

Składnia: ILORAZC n n → n

Opis działania: funkcja, której wartością jest część całkowita ilorazu pierwszego parametru przez drugi.

Parametry: dwa wyrażenia arytmetyczne. Wartość drugiego parametru musi być różna od zera.

Uwaga: wartości parametrów nie są zaokrąglane przed wykonaniem dzielenia.

Wynik: liczba calkowita.

Przykłady

```
?pisz ilorazc 16 3
5
?pisz ilorazc 7 3.6
1
?pisz ilorazc -4 3
-1
```

INACZEJ **ELSE**

Element składni procedury JEŚLI (ang. IF).

JASKRAWO **BRIGHTON (BON)**

Składnia: JASKRAWO

Opis działania: wykonanie procedury nie pociąga za sobą żadnych skutków. Procedura ta została zachowana ze względu na zgodność z Junior–LOGO, gdzie wybierała ona jaskrawą paletę kolorów. Brak palet kolorów w AC–LOGO jest zrównoważony większą liczbą dostępnych kolorów.

Uwaga: procedura "pusta"

JEŚLI **IF**

Składnia:

```
JEŚLI b l
lub: JEŚLI b l INACZEJ l
```

oraz w wyrażeniach:

```
JEŚLI b l → sl
lub: JEŚLI b l INACZEJ l → sl
```

Opis działania: procedura warunkowa, która użyta w wyrażeniu jest funkcją. Jej pierwszym parametrem jest wyrażenie o wartości logicznej. Jeśli wartością tego wyrażenia jest PRAWDA, czyli jeśli jest spełniony podany warunek, to jest wykonywana lista procedur

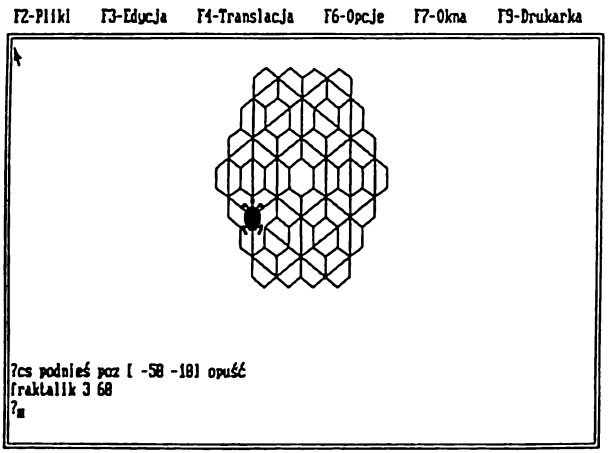
będąca drugim parametrem. W drugiej postaci procedury JEŚLI, dodatkowo, gdy podany warunek nie jest spełniony, czyli gdy wartością wyrażenia jest FAŁSZ, to jest wykonywana lista procedur podana po słowie INACZEJ (ang. ELSE).

Parametry: dwa lub trzy wyrażenia, w zależności od postaci procedury. Pierwsze wyrażenie ma wartość logiczną; wartością drugiego i trzeciego wyrażenia jest lista, której elementami są wywołania procedur i ich parametry aktualne. W postaci funkcyjnej procedury JEŚLI elementami tych list są wyrażenia.

Wynik (tylko dla postaci funkcyjnej): słowo lub lista.

Przykłady

```
?przypisz "wiek_ali 15
?pisz jeśli :wiek_ali < 18 ["dziecko] inaczej ["dorosła]
dziecko
?przypisz "wiek_ali 21
?jeśli :wiek_ali < 18 [pisz "dziecko] inaczej [pisz "dorosła]
dorosła
```



Rys. 8.5. Efekt wykonania procedury FRAKTALIK

```
?oto fraktalik :poziom :bok
> jeśli :poziom = 0 [stop]
> powtórz 6
> [
```

```
> naprzód :bok
> fraktalik :poziom — 1 :bok / 2
> prawo 60
> ]
>już
FRAKTALIK zdefiniowano
?czyść
?podnieś
?poz [—50 —10]
?opuść
?fraktalik 3 40
```

JUŻ **END**

Składnia: JUŻ

Opis działania: ostatnie słowo treści każdej procedury — porównaj opis procedury OTO (ang. TO).

KAT

Patrz KATALOG

KATALOG **CAT**

Składnia: KATALOG
lub: KATALOG s
w analizie wyrażenia: KATALOG → l
lub: KATALOG s → l

Opis działania: procedura lub funkcja wyświetlania katalogu dysku w formie skróconej.

Parametr: słowo składające się z (opcjonalnie) nazwy dysku, (opcjonalnie) nazwy katalogu, (opcjonalnie) nazwy pliku (jednoznacznej lub niejednoznacznej). W przypadku braku parametru jest wyświetlany katalog dysku bieżącego.

Wynik: jeśli funkcja jest wywołana podczas analizy wyrażenia, to wynikiem jej działania jest lista zawierająca słowa odpowiadające nazwom wszystkich wybranych plików.

Uwaga: parametr musi być pojedynczym słowem.

Przykład

```
?katalog
Katalog C:\LOGO\LOGO.POL:
ACLOGO    EXE : SYSTEM    FNT : LAT8x14.FNT : LOGO.CFG
?katalog "b:ala.exe
Katalog jest pusty lub nie istnieje.
?katalog "b:
Katalog B:\
ALA       LOG : DOMEK     LOG : OLA     LOG : L       LOG
ELA       LOG : FRAKT1    LOG : IZA     LOG
?katalog "b:*.*
Katalog B:\
ALA       LOG : DOMEK     LOG : OLA     LOG : L       LOG
ELA       LOG : FRAKT1    LOG : IZA     LOG
?katalog "b:???.*
Katalog B:\
ALA       LOG : OLA       LOG : ELA     LOG : IZA.    LOG
LOG       TXT : ALA       TXT
?pokaż sortuj katalog "b:???.*
[ ALA.LOG ALA.TXT ELA.LOG IZA.LOG LOG.TXT OLA.LOG ]
```

KĄT — SETHEADING (SETH)

Składnia: KĄT n

Opis działania: procedura ustawiająca żółwia pod kątem podanym jako parametr.

Parametr: wyrażenie, którego wartością jest kąt w stopniach mierzony od pionu zgodnie z ruchem wskazówek zegara.

Przykłady

```
?czyść
?kąt 380
?pisz kąt?
20
?kąt azymut [20 20]
?pisz kąt?
45
{prosty przykład animacji komputerowej}
?oto pułapka_na_żółwie :rozmiar :tempo
> wróć grafika zamaluj pż
> opisokna (zdanie –:rozmiar –:rozmiar :rozmiar :rozmiar)
> okno zmażokno
> przypisz "rozmiar :rozmiar – 10
> podnieś
```

```
) taniec
)już
PUŁAPKA_NA_ŻÓŁWIE zdefiniowano
?oto taniec
) jeśli lub (:rozmiar < abs pozx?)
)           (:rozmiar < abs pozy?)
)  [nuta [1 1] wstecz :tempo kąt losowa 360]
) naprzód :tempo
) taniec
)już
TANIEC zdefiniowano
?pulapka_na_żółwie 41 10
```

KĄT? **HEADING**

Składnia: KĄT? → n

Opis działania: funkcja, której wartością jest kąt między pionem a aktualnym kierunkiem żółwia.

Wynik: kąt w stopniach.

Przykład

```
?czyść
?pisz kąt?
0
?prawo 410
?pisz kąt?
50
```

KLAWISZ? **KEYPRES (KP)**

Składnia: KLAWISZ → b

Opis działania: funkcja o wartości logicznej PRAWDA, jeśli w trakcie wykonywania programu wprowadzono dowolny znak z klawiatury, który nie został odczytany za pomocą funkcji CZYTAJZNAK (ang. READCHAR) lub CZYTAJLISTĘ (ang. READLIST).

Wynik: wartość logiczna.

Uwaga: naciśnięcie klawiszy: CTRL, SHIFT, ALT oraz CAPS LOCK nie powoduje wczytania znaku.

Przykład

```
?powtórz 10 [jeśli klawisz? [ps "tak] inaczej [pisz "nie]]
```

```
nie
nie
nie
nie
nie
tak         {przyciśnięty klawisz}
tak
tak
tak         {zwolniony klawisz}
tak
```

KOLORP? **POINTCOLOR**

Składnia: KOLORP? l → n

Opis działania: funkcja podająca aktualny kolor punktu o współrzędnych (x,y) pobranych z listy l. Zakres parametru wyjściowego funkcji jest zależny od posiadanej karty graficznej: dla karty EGA/VGA wynik zawiera się w granicach od 0 (kolor "czarny") do 15 (kolor "biały"), natomiast dla monochromatycznej karty HGC parametr jest równy 0 dla punktów ciemnych i 15 dla punktów jasnych.

Wynik: wartość z zakresu od 0 do 15 (kolorowa karta graficzna) lub 0 albo 15 (monochromatyczna karta graficzna).

Uwaga: dla punktów o współrzędnych poza ekranem funkcja zawsze podaje wynik równy 0

Przykład

```
?cs ps kolorp? [ 0 0 ]
0
?naprzód 1 ps kolorp? [ 0 0 ]
1
```

KOLORTEKSTU (KT) **TEXTCOLOR (TC)**

Składnia: KOLORTEKSTU n

Opis działania: procedura ustalająca kolor znaków pisanych w oknie graficzno-tekstowym.

Parametr: wyrażenie, którego wartością jest liczba z zakresu (0, 15), będąca numerem koloru. Liczby niecałkowite są zaokrąglane. Liczby większe od 15 są niejawnie przekształcane do zakresu (0, 15) za pomocą operacji modulo 16.

Przykład
?kolortekstu 2
{Kolor następnych drukowanych znaków zmienia się na zielony}

KT

Patrz KOLORTEKSTU

KURSOR **SETCURSOR**

Składnia: KURSOR l

Opis działania: procedura ustawiająca kursor tekstowy w miejscu określonym przez parametr.

Parametr: wyrażenie, którego wartością jest dwuelementowa lista postaci [n n]. Pierwszym elementem listy jest numer kolumny (większy lub równy zero). Drugim elementem listy jest numer wiersza (także większy lub równy zero). Jeśli numer kolumny lub numer wiersza wykraczają poza rozmiary okna tekstowego, to kursor jest umieszczany w maksymalnej możliwej pozycji.

Uwagi: pozycja kursora o współrzędnych [0 0] oznacza lewy, górny róg okienka graficzno-tekstowego. Miejsce na ekranie, w którym znajdzie się kursor zależy od aktualnego generatora znaków (jego wielkości w poziomie i w pionie).

Przykład
?teksty
?kursor [35 12] pisz [To jest tekst na środku okienka]
{Wydrukowanie tekstu w centrum okienka, poczynając od kolumny 35 w wierszu 12}

KURSOR? **CURSOR**

Składnia: KURSOR? l

Opis działania: funkcja, której wartością jest lista zawierająca aktualne współrzędne kursora tekstowego.

Wynik: dwuelementowa lista postaci [n n]. Pierwszym elementem listy jest numer kolumny, a drugim numer wiersza aktualnej pozycji kursora. Numery te zależą od aktualnego generatora znaków.

Uwaga: pozycja kursora o współrzędnych [0 0] oznacza lewy, górny róg okienka graficzno-tekstowego.

Przykład
```
?teksty
?pisz kursor?
0 1
```

LEWO (LW) **LEFT (LT)**

Składnia: LEWO n

Opis działania: procedura obracająca żółwia w lewo o podany kąt.

Parametr: wyrażenie, którego wartością jest kąt w stopniach.

Uwaga: podanie ujemnej wartości kąta powoduje obrót w prawo.

Przykład
```
?czyść
?lewo 410
?pisz kąt?
310
```

LEWYPRZYCISK (LPM) **MOUSELEFTBUTTON (MLB)**

Składnia: LEWYPRZYCISK → b

Opis działania: funkcja o wartości logicznej PRAWDA, jeśli w momencie wywołania procedury był naciśnięty lewy przycisk myszki.

Wynik: wartość logiczna.

Uwaga: poprzednie naciśnięcia klawiszy myszki nie są pamiętane (w odróżnieniu od funkcji KLAWISZ?).

Przykład
```
?powtórz 8 [jeśli lewyprzycisk [ps "tak] inaczej [pisz "nie]]
nie
nie
nie
tak        {przyciśnięty lewy przycisk myszki}
tak
tak
nie        {lewy przycisk zwolniony}
nie
```

LICZBA? **NUMBERP**

Składnia: LICZBA? sl → b

Opis działania: funkcja, której wartością jest PRAWDA, jeśli jej parametr jest liczbą, lub FAŁSZ w przeciwnym razie.

Parametr: dowolne wyrażenie.

Wynik: wartość logiczna.

Przykłady

```
?pisz liczba? 2.39
PRAWDA
?pisz liczba? 5 + 8 * 1.1
PRAWDA
?pisz liczba? [239]
FAŁSZ
?pisz liczba? "3781
PRAWDA
?pisz liczba? 3 > 1
FAŁSZ
?oto sześcian :x
> jeśli liczba? :x
>   [ wynik :x * :x * :x]
> inaczej
>   [pisz "Błąd przerwij]
>już
SZEŚCIAN zdefiniowano
?pisz sześcian 2
8
?pisz sześcian "kwadrat
Błąd
*** Przerwanie *** na poziomie 1 w wierszu 5 procedurze SZEŚCIAN
```

LISTA **LIST**

Składnia: LISTA sl sl → l
lub: (LISTA sl sl ...) → l

Opis działania: funkcja, której wartością jest lista utworzona z jej parametrów przez dodanie zewnętrznych nawiasów.

Parametry: dwa dowolne wyrażenia. W wersji zachłannej — od 2 do 127 dowolnych wyrażeń.

Wynik: lista, której elementami są parametry.

Uwaga: jeśli parametr jest listą, to staje się on podlistą na liście wynikowej.

Przykłady

```
?pokaż lista 1 2
[ 1 2 ]
?pokaż lista [] 1
[ [ ] 1 ]
?pokaż lista "Ala "Ola
[ Ala Ola ]
?pokaż (lista "Ala "Ola "Jola "Pola)
[ Ala Ola Jola Pola]
?pokaż lista lista [Ala] [Ola] [Jan]
[ [ [ Ala ] [ Ola ] ] [ Jan ] ]
?pokaż (lista [Ala] [Ola] [Jan])
[ [ Ala ] [ Ola ] [ Jan ] ]
```

LISTA? **LISTP**

Składnia: LISTA? sl → b

Opis działania: funkcja, której wartością jest PRAWDA, jeśli jej parametr jest listą, lub FAŁSZ w przeciwnym razie.

Parametr: dowolne wyrażenie.

Wynik: wartość logiczna.

Przykłady

```
?pisz lista? 1 + 2
FAŁSZ
?pisz lista? "Ala
FAŁSZ
?pisz lista? [[] Ola]
PRAWDA
?pisz lista? pierw [[] Ola]
PRAWDA
?pisz lista? 2 > 1
FAŁSZ
```

LN **LN**

Składnia: LN n → n

Opis działania: funkcja, której wartością jest logarytm naturalny z parametru.

Parametr: wyrażenie arytmetyczne o wartości dodatniej.

Wynik: logarytm naturalny parametru.

Przykłady

```
?pisz ln 1
0
?pisz ln 10
2.302585092994046
?pisz ln exp 1.5
1.5
?pisz exp ln 1.5
1.5
```

LOKALNA — LOCAL

Składnia: LOKALNA s
lub: (LOKALNA s s ...)

Opis działania: deklaracja zmiennej lokalnej procedury. Parametr określa nazwę zmiennej. Zmiennej nie jest nadawana wartość początkowa.

Parametr: wyrażenie, którego wartością jest słowo będące nazwą zmiennej lokalnej. Nazwa ta musi być różna od nazw parametrów formalnych procedury, w której zmienna jest deklarowana. W wersji zachłannej — dowolna, większa od jeden, liczba wyrażeń.

Przykład

```
?oto dziewczę :imię
{imię jest parametrem formalnym procedury dziewczę}
> lokalna "imienniczka
{imienniczka jest nazwą zmiennej lokalnej procedury dziewczę}
> przypisz "imienniczka :imię
> pisz :imienniczka
>już
DZIEWCZĘ zdefiniowano
?dziewczę "Ala
Ala
?pisz :imienniczka
Zmienna IMIENNICZKA nie ma zdefiniowanej wartości.
{imienniczka nie jest nazwą zmiennej poza procedurą dziewczę}
```

LOSOWA — RANDOM

Składnia: LOSOWA n → n

Opis działania: funkcja generująca liczbę pseudolosową.

Parametr: wyrażenie, którego wartością jest liczba dodatnia.

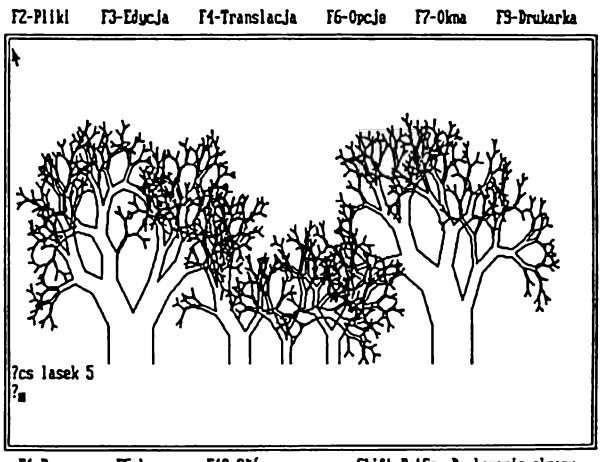
Wynik: liczba o wartości pseudolosowej z zakresu od zera do wartości parametru.

Przykłady

```
?powtórz 5 [pisz losowa 5]
{generowanie wartości pseudolosowych z przedziału (0, 5)}
0.0526444288460952
0.0196844386120182
1.67561876277963
0.166173284096805
1.77846613971374
?powtórz 3 [pisz -losowa 5]
{generowanie wartości pseudolosowych z przedziału (-5, 0)}
-1.08584856715598
-2.68471327860347
-0.97872859889523
?powtórz 5 [pisz ( losowa 7 ) - 4]
{generowanie wartości pseudolosowych z przedziału (-4, 3)}
0.902157658619953
2.64922025208289
-2.07669301431318
-0.890194402905362
-3.23734244819483
?oto lasek :iledrzew
> okno czyść sż
> podnieś pozx -120
> powtórz :iledrzew
> [
>   podnieś pozy -65
>   pozx pozx? + ( 80 + losowa 200 ) / (1 + :iledrzew)
>   opuść drzewo 15 + losowa 15
> ]
>już
LASEK zdefiniowano
?oto drzewo :rozmiar
> ( lokalna "rozłożystość "gałąź )
> przypisz "rozłożystość 15 + losowa 30
> przypisz "gałąź .6 + losowa .35
> jeśli :rozmiar < 5
```

```
    [ prawo 90 naprzód 0.3 lewo 90 stop ]
> naprzód :rozmiar
> lewo :rozłożystość
> drzewo :gałąź * :rozmiar
> prawo 2 * :rozłożystość
> drzewo :gałąź * :rozmiar
> lewo :rozłożystość
> wstecz :rozmiar
>już
DRZEWO zdefiniowano
?lasek 5
```

{efekt wykonania procedury LASEK pokazano na rys. 8.6}



Rys. 8.6. Efekt wykonania procedury LASEK

LOSUJ **RERANDOM**

Składnia: LOSUJ n

Opis działania: procedura inicjująca generator liczb pseudolosowych.

Parametr: wyrażenie o dodatniej wartości liczbowej. Użycie tej samej wartości n spowoduje wygenerowanie zawsze takiego samego ciągu liczb pseudolosowych.

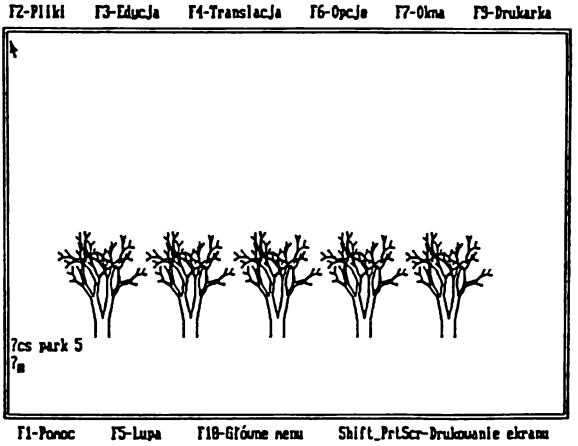
```
Przykłady
    ?losuj 2
    ?powtórz 3 [pisz losowa 10]
    0.210882900479141
    9.97375408185064
    6.66310617389447

    ?losuj 5
    ?powtórz 3 [pisz losowa 10]
    0.527970213934751
    9.7766045106357
    6.59840693380535

    ?losuj 2
    ?powtórz 3 [pisz losowa 10]
    0.210882900479141
    9.97375408185064
    6.66310617389447

    ?losuj 5
    ?powtórz 3 [pisz losowa 10]
    0.527970213934751
    9.7766045106357
    6.59840693380535

    ?oto park :iledrzew
    > okno czyść sż
    > podnieś pozx -120
    > powtórz :iledrzew
    > [
    >   podnieś pozy -65
    >   pozx pozx? + 200 / (1 + :iledrzew)
    >   losuj 7
    >   opuść drzewo 20
    > ]
    >już
    PARK zdefiniowano
    {Deklaracja procedury drzewo — w przykładzie do procedury LOSOWA}
    ?park 5 {por. przykład użycia procedury LOSOWA}
```



Rys. 8.7. Efekt wykonania procedury PARK

LPM

Patrz LEWYPRZYCISK

LUB **OR**

Składnia: LUB b b → b
lub: (LUB b b ...) → b

Opis działania: funkcja, której wartością jest suma logiczna parametrów.

Parametry: dwa wyrażenia o wartościach logicznych. W wersji zachłannej — dowolna, większa niż jeden liczba wyrażeń o wartościach logicznych.

Wynik: suma logiczna parametrów.

Przykłady

```
?pisz lub falsz falsz
FAŁSZ
?pisz lub falsz prawda
PRAWDA
?pisz lub prawda prawda
PRAWDA
```

```
?pisz (lub 2 * 2 = 3  2 * 2 = 4  2 * 2 = 5)
PRAWDA
?oto owoc? :x
> wynik (lub :x = "jablko :x = "śliwka :x = "gruszka)
>już
OWOC? zdefiniowano
?pisz owoc? "jablko
PRAWDA
?pisz owoc? "Liliana
FAŁSZ
```

LW

Patrz LEWO

ŁADUJ **LOAD**

Składnia: ŁADUJ s

Opis działania: procedura powodująca załadowanie z dysku do pamięci operacyjnej pliku o nazwie podanej jako parametr. Plik powinien zawierać teksty źródłowe procedur.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeśli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie LOG (lub inne standardowe rozszerzenie ustalone w pliku konfiguracyjnym).

Uwaga: efektem wykonania procedury ŁADUJ może być zredefiniowanie istniejącej procedury. Warto zwrócić uwagę na szczególny przypadek, gdy wywołanie ŁADUJ z wnętrza procedury powoduje zredefiniowanie tej procedury. Wówczas aż do jej zakończenia, czyli do wykonania procedury pierwotnej JUŻ, jest wykonywana jej "stara" treść. Treść ta nie jest jednak dostępna za pomocą procedury pierwotnej TREŚĆ (ang. TEXT), która bezpośrednio po wykonaniu procedury ŁADUJ udostępnia "nową" treść deklarowanej procedury.

Przykłady

```
?ładuj "b:rysunki.log
{Załadowanie z dysku B: do pamięci operacyjnej pliku RYSUNKI.LOG}
?ładuj "domek
{Załadowanie z dysku i z katalogu bieżącego do pamięci operacyjnej pliku
DOMEK.LOG }
```

```
{Realizacja procedur nakładkowych za pomocą procedury pierwotnej ŁADUJ}
    ?oto demo
    > ps 0
    >już
    DEMO zdefiniowano
    ?zapisz "proc0
    ?oto demo
    > ps 1
    >już
    DEMO zdefiniowano
    ?zapisz "proc1
    ?oto demo
    > ps 2
    >już
    DEMO zdefiniowano
    ?zapisz "proc2
    ?oto demo
    > _demo 0
    >już
    DEMO zdefiniowano
    ?oto _demo :i
    > ładuj slowo "proc :i
    > demo
    > _demo reszta :i + 1 3
    >już
    _DEMO zdefiniowano
    ?demo
    DEMO zdefiniowano
    0
    DEMO zdefiniowano
    1
    DEMO zdefiniowano
    2
    DEMO zdefiniowano
    0
```

ŁADUJO **LOADSCR**

Składnia: ŁADUJO s

Opis działania: procedura "ładuj obraz", powodująca załadowanie z pamięci zewnętrznej do pamięci ekranu (w obszarze okienka graficzno-tekstowego), zawartości pliku o nazwie podanej jako parametr.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeśli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie SCR (rozszerzenie to można ustalić na inne w pliku konfiguracyjnym).

Przykłady

```
?ładujo "ekran
{Zaladowanie do pamięci ekranu zawartości pliku EKRAN.SCR znajdującego
się na dysku bieżącym}
?ładujo "b:obrazek.cod
{Zaladowanie do pamięci ekranu zawartości pliku OBRAZEK.COD
znajdującego się na dysku B: }
```

ŁADUJOKNO **LOADWND**

Składnia: ŁADUJOKNO s

Opis działania: procedura "ładuj okno", powodująca załadowanie z pamięci zewnętrznej do pamięci ekranu (w obszarze okienka graficznego), zawartości pliku o nazwie podanej jako parametr.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeśli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie SCR (rozszerzenie to można ustalić na inne w pliku konfiguracyjnym).

Przykład

```
?ładujokno "ekran
{Zaladowanie do okienka graficznego zawartości pliku EKRAN.SCR
znajdującego się na dysku bieżącym}
?opisokna [ -50 -50 50 50 ] ładujokno "b:obrazek.cod
{Zaladowanie do okienka graficznego zawartości pliku OBRAZEK.COD
znajdującego się na dysku B: }
```

ŁADUJRED **LOADED**

Składnia: ŁADUJRED s

Opis działania: procedura "ładuj obszar edytora", powodująca załadowanie z pamięci zewnętrznej do obszaru roboczego edytora zawartości pliku tekstowego o nazwie podanej jako parametr.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeśli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie LOG (rozszerzenie to można ustalić na inne w pliku konfiguracyjnym).

Uwaga: załadowany do obszaru roboczego edytora tekst nie podlega automatycznej kompilacji. Tekst ten można redagować po wydaniu polecenia RED bez parametrów lub po wywołaniu edytora z systemu okienkowego.

Przykład

```
?ładujred "przyklad
{Załadowanie do obszaru roboczego edytora zawartości pliku PRZYKLAD:LOG
znajdującego się na dysku bieżącym}
?ed
{wejście do edytora i poprawienie załadowanego pliku, wyjście przez polecenie
Ctrl–K D edytora}
PRZYKLAD zdefiniowano.
?
```

ŁADUJW **LOADOBJ**

Składnia: ŁADUJW s → nsl

Opis działania: funkcja "ładuj wartość", której wartością jest lista powstała z tekstu załadowanego z dysku, z pliku o nazwie będącej parametrem. Lista ta jest tworzona przez uzupełnienie tekstu zewnętrznymi nawiasami kwadratowymi.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeśli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie TXT (rozszerzenie to można zmienić w pliku konfiguracyjnym).

Przykłady

```
?zapiszw."ala [Ala ma kota]
{ALA jest nazwą podstawową pliku o pełnej nazwie ALA.TXT; na dysku jest
zapisywany plik tekstowy: Ala ma kota}
?pisz ładujw "ala.txt
Ala ma kota
?przypisz "ola ładujw "ala
{Przypisanie zmiennej ola listy utworzonej z tekstu zawartego w pliku ALA.TXT,
który jest przechowywany na dysku bieżącym}
?pisz :ola
Ala ma kota
?zapiszw "ala "Ala
{ALA jest nazwą podstawową pliku o pełnej nazwie ALA.TXT; Na dysku jest
zapisywany plik tekstowy zawierający znaki "Ala" }
?pokaż ładujw "ala
[ Ala ]
```

```
{Realizacja nakładkowych procedur PROC1 i PROC2 za pomocą procedur
pierwotnych ŁADUJW, OKREŚL i USUŃ}
?zapiszw "proc1 [[][pisz [To jest procedura nr. 1] już]]
?zapiszw "proc2 [[][pisz [To jest procedura nr. 2] już]]
?oto program
> lokalna "znak
>
> przypisz "znak czytajznak
> jeśli równe? :znak "s [stop]
> jeśli równe? :znak "1
> [
>   jeśli nie określone? "proc1
>     [usuń "proc2 określ "proc1 ładujw "proc1]
>   proc1
> ]
> jeśli równe? :znak "2
> [
>   jeśli nie określone? "proc2
>     [usuń "proc1 określ "proc2 ładujw "proc2]
>   proc2
> ]
> program
>już
PROGRAM zdefiniowano
?program
{naciśnięcie klawiszá 1 — odczyt pliku PROC1.TXT z dysku}
To jest procedura nr. 1
{naciśnięcie klawisza 1}
To jest procedura nr. 1
{naciśnięcie klawisza 2 — odczyt pliku PROC2.TXT z dysku}
To jest procedura nr. 2
{naciśnięcie klawisza 2}
To jest procedura nr. 2
```

MARGINESYDRUKU **SETCOPYMARGIN**

Składnia: MARGINESYDRUKU l

Opis działania: procedura ustala marginesy wydruku kopii ekranu lub tekstu edytora na drukarce.

Parametr: lista definiująca górny margines wydruku (liczbę pustych wierszy na górze każdej strony wydruku) oraz lewy margines (liczbę spacji drukowanych na początku każdego wiersza).

Przykłady

```
?marginesydruku [ 0 0 ]
{wydruk będzie realizowany od lewego, górnego rogu każdej strony}
?marginesydruku [ 1 10 ]
{na początku każdej strony wydruku będzie zostawiony jeden pusty wiersz;
każdy wiersz wydruku będzie poprzedzony wydrukowaniem dziesięciu spacji
(ok. 2.5 cm marginesu)}
```

MIGAJĄCO **FLASHON (FON)**

Składnia: MIGAJĄCO

Opis działania: wykonanie procedury nie pociąga za sobą żadnych skutków. Procedura ta została zachowana ze względu na zgodno z Junior–LOGO.

Uwaga: procedura "pusta"

ML **LOWERCASE (LC)**

Składnia: ML s → s

Opis działania: funkcja "małe litery", której wartością jest słowo powstałe z parametru przez zamianę wielkich liter na małe.

Parametr: wyrażenie, którego wartością jest słowo.

Wynik: słowo nie zawierające wielkich liter.

Uwaga: w wyniku działania funkcji nie ulegają zmianie znaki nie będące literami, natomiast są zamieniane litery alfabetów narodowych.

Przykłady

```
?pisz ml "Ala
ala
?pisz ml "PĄCZUSZEK
pączuszek
?pisz ml "ZNAKII"#$%&'
znakii"#$%&'
?pisz ml 123
123
?pisz ml "To jest jedno słowo'
to jest jedno słowo
```

NAK **LPUT**

Składnia: NAK sl l → l

Opis działania: funkcja "na koniec", której wartością jest lista utworzona przez dołączenie pierwszego parametru na koniec listy będącej drugim parametrem.

Parametry: pierwszy parametr jest wyrażeniem, którego wartością jest słowo lub lista. Drugi parametr jest wyrażeniem, którego wartością jest lista.

Wynik: lista.

Przykłady

```
?pokaż nak "Jan [Ala Ola]
[ Ala Ola Jan ]
?pokaż nak [Jan] [Ala Ola]
[ Ala Ola [Jan] ]
?pokaż nak 2 [Ala Ola]
[ Ala Ola 2 ]
?pokaż nak [] []
[ [ ] ]
```

NAP **FPUT**

Składnia: NAP sl l → l

Opis działania: funkcja "na początek", której wartością jest lista utworzona przez dołączenie pierwszego parametru na początek listy będącej drugim parametrem.

Parametry: pierwszy parametr jest wyrażeniem, którego wartością jest słowo lub lista. Drugi parametr jest wyrażeniem, którego wartością jest lista.

Wynik: lista.

Przykłady

```
?pokaż nap "Jan [Ala Ola]
[ Jan Ala Ola ]
?pokaż nap [Jan] [Ala Ola]
[ [Jan] Ala Ola ]
?pokaż nap 2 [Ala Ola]
[ 2 Ala Ola ]
?pokaż nap [] []
[ [ ] ]
```

NAPRZÓD (NP) **FORWARD (FD)**

Składnia: NAPRZÓD n

Opis działania: procedura przesuwająca żółwia o zadaną liczbę kroków do przodu.

Parametr: wyrażenie arytmetyczne określające liczb kroków z zakresu od –32000 do 32000.

Uwagi: przy ujemnej wartości parametru następuje przesunięcie żółwia do tyłu. Standardowo krokiem żółwia jest odległość między dwoma kolejnymi punktami ekranu, zarówno w pionie jak i poziomie. Krok można zmieniać za pomocą procedury PROPORCJA (ang. SETSCRUNCH).

Przykłady

```
?naprzód 20 * 2
{Przesunięcie żółwia o 40 kroków do przodu}
?naprzód –50
{Przesunięcie żółwia o 50 kroków do tyłu}
```

NIE **NOT**

Składnia: NIE b → b

Opis działania: funkcja, której wartością jest zaprzeczenie logiczne parametru.

Parametr: wyrażenie logiczne.

Wynik: wartość logiczna będąca zaprzeczeniem parametru.

Przykłady

```
?pisz nie prawda
FAŁSZ
?pisz nie falsz
PRAWDA
?przypisz "elementarz [Ala ma kota]
?oto drukuj_listę :lista
>jeśli nie puste? :lista
> [pisz :lista]
>inaczej
> [pisz [Błąd — lista jest pusta]]
>już
DRUKUJ_LISTĘ zdefiniowano
?drukuj_listę :elementarz
Ala ma kota
?drukuj_listę []
 Błąd — lista jest pusta
```

NIEJASKRAWO **BRIGHTOFF (BOF)**

Składnia: NIEJASKRAWO

Opis działania: wykonanie procedury nie pociąga za sobą żadnych skutków. Procedura ta została zachowana ze względu na zgodność z Junior–LOGO.

Uwaga: procedura "pusta"

NIEMIGAJĄCO **FLASHOFF (FOF)**

Składnia: NIEMIGAJĄCO

Opis działania: wykonanie procedury nie pociąga za sobą żadnych skutków. Procedura ta została zachowana ze względu na zgodność z Junior–LOGO.

Uwaga: procedura "pusta"

NP

Patrz NAPRZÓD

NUTA **SOUND**

Składnia: NUTA l

Opis działania: procedura generująca dźwięki.

Parametr: wyrażenie, którego wartością jest dwuelementowa lista postaci [n n]. Pierwszy element listy jest liczbą z zakresu (1, 119) określającą wysokość dźwięku w półtonach, natomiast drugi — liczbą dodatnią określającą czas trwania dźwięku w 1/100 s. Wartość 47 pierwszego elementu listy odpowiada dźwiękowi a^1 ("la") o częstotliwości 440 Hz.

Przykład

```
?oto gama :tempo
{:tempo jest czasem trwania dźwięku}
> (lokalna "i "gama)
> przypisz "gama      [38 40 42 43 45 47 49 50]
>                      {c  d  e  f  g  a  h  c}
> przypisz "i 0
```

```
> powtórz 8
> [
>   przypisz "i :i + 1
>   nuta zdanie element :i :gama :tempo
> ]
> powtórz 7
> [
>   przypisz "i :i – 1
>   nuta zdanie element :i :gama :tempo
> ]
>już
GAMA zdefiniowano
?gama 24 gama 6 gama 12
?gama 1
```

ODLICZANIE **STARTTIMER**

Składnia: ODLICZANIE

Opis działania: procedura włącza odliczanie czasu dla stopera. Czas ten można następnie odczytać za pomocą funkcji STOPER (ang. TIMER).

Uwagi: stoper jest włączany po raz pierwszy w momencie uruchomienia programu LOGO — w przypadku nieużycia procedury ODLICZANIE funkcja STOPER podaje czas działania programu LOGO.

Przykłady

```
?ps stoper
121.23
{czas działania programu LOGO w sekundach}
?odliczanie ps stoper
0
```

ODŚMIEĆ **RECYCLE**

Składnia: ODŚMIEĆ

Opis działania: procedura oczyszczająca pamięć roboczą przez jej uporządkowanie i odzyskanie nieużytków.

Uwagi: jeżeli w trakcie wykonywania programu cała pamięć operacyjna zostanie zajęta, to wówczas procedura odzyskująca nieużytki będzie wywołana samoczynnie. Odzyskiwanie nieużytków jest sygnalizowane za pomocą animacji znaku kreski w lewym górnym rogu ekranu. Zmiany sposobu sygnalizacji lub jej

wyłączenia można dokonać za pomocą procedury pierwotnej .ODŚMIECANIE (ang. .INSTALL). Procedura ta umożliwia też modyfikację parametrów procedury odzyskiwania nieużytków.

ODWR

Patrz ODWRACANIE

ODWRACANIE (ODWR) **PENREVERSE (PX)**

Składnia: ODWRACANIE

Opis działania: procedura wprowadzająca pisak żółwia w stan "odwracanie". W tym stanie pisak, w trakcie przesuwania żółwia, zmienia kolory punktów na kolory dopełnieniowe (np. kolor 0 na 15, 1 na 14, 2 na 13, itd).

Uwaga: wykonanie procedury: OPUŚĆ (ang. PENDOWN), PODNIEŚ (ang. PENUP) lub ŚCIERANIE (ang. PENERASE) kończy efekt działania procedury ODWRACANIE. Natomiast wykonanie procedury CZYŚĆ (ang. CLEARSCREEN) i ZMAŻ (ang. CLEAN) nie wpływa na jej działanie.

Przykłady

```
?czyść naprzód 20
{Żółw rysuje linię pionową o długości 20 kroków}
?podnieś naprzód 20
{Żółw przesuwa się w pionie o 20 kroków, nie pozostawiając za sobą śladu}
?odwracanie wstecz 40
{Żółw rysuje linię pionową o długości 20 kroków oraz maże poprzednio
narysowaną linię}
?czyść powtórz 10 [naprzód 4 ]
{Żółw rysuje pionową linię przerywaną o długości 40 kroków}
```

OKNO **WINDOW**

Składnia: OKNO

Opis działania: procedura przejścia do takiego trybu okna graficznego, w którym żółw po wyjściu poza okno graficzne nadal rysuje pomimo, że jego ślad jest niewidoczny. Efekt działania procedury OKNO kończy się po wykonaniu procedury PŁOT (ang. FENCE) lub SKLEJ (ang. WRAP).

Przyklad
?czyść
?okno
?prawo 10 naprzód 1000
{Żółw znajduje się poza oknem graficznym}

OKREŚL **DEFINE**

Składnia: OKREŚL s l

Opis działania: procedura tworząca postać półskompilowaną procedury o nazwie będącej pierwszym parametrem oraz o parametrach formalnych i treści zawartej w liście będącej drugim parametrem.

Parametry: wartością pierwszego parametru jest słowo będące nazwą deklarowanej procedury. Wartością drugiego parametru jest lista złożona z dwóch podlist. Elementami pierwszej podlisty są parametry formalne, a drugiej — treść deklarowanej procedury. Jeżeli procedura nie ma parametrów formalnych, to pierwsza podlista jest pusta.

Uwagi: procedury zadeklarowanej za pomocą procedury OKREŚL nie można poprawiać za pomocą procedur RED (ang. EDIT) i REDW (ang. EDALL) ani oglądać za pomocą procedur POW (ang. POALL), POWP (ang. POPS) i PO (ang. PRINTOUT), chociaż można ją kasować procedurą USUŃ (ang. ERASE). Wynika to z tego, że procedura OKREŚL tworzy wyłącznie postać półskompilowaną deklarowanej procedury. Tekst źródłowy procedury zadeklarowanej za pomocą procedury OKREŚL, można odtworzyć z tekstu półskompilowanego za pomocą procedury TREŚĆ (ang. TEXT). Można także zapamiętać treść deklarowanej procedury w zmiennej.
Warto zwrócić uwagę na szczególny przypadek, gdy wywołanie OKREŚL z wnętrza procedury powoduje zredefiniowanie tej procedury. Wówczas aż do jej zakończenia, czyli do wykonania procedury pierwotnej JUŻ, jest wykonywana jej "stara" treść. Treść ta nie jest jednak dostępna za pomocą procedury pierwotnej TREŚĆ, która bezpośrednio po wykonaniu procedury OKREŚL, udostępnia "nową" treść deklarowanej procedury. Deklarowanie procedur za pomocą OKREŚL umożliwia bardziej oszczędną gospodarkę pamięcią niż deklarowanie procedur za pomocą OTO (ang. TO), co jest przydatne w długich programach. Ponadto, procedura OKREŚL wraz z funkcją ŁADUJW (ang. LOADOBJ) umożliwia tworzenie procedur nakładkowych (por. przykład do funkcji ŁADUJW).

Przykłady

```
?określ "funkcja [[:i][wynik (sin :i) + (cos :i) już]]
?pisz funkcja 0
1
?pisz treść "funkcja
[ :i ] [ wynik (sin :i) + (cos :i) już ]
?przypisz "tekst_drukuj [[:i][ {komentarz} pisz :i już]]
?określ "drukuj :tekst_drukuj
{Zdefiniowanie odpowiednika procedury PISZ}
?drukuj "Iwona
Iwona
?pisz :tekst_drukuj
[ :i ] [ pisz :i już ]
?pisz treść "drukuj
[ :i ] [ pisz :i już ]
```

OKREŚLONE? **DEFINEDP**

Składnia: OKREŚLONE? s → b

Opis działania: funkcja, której wartością jest PRAWDA, jeśli jej parametr jest nazwą procedury zadeklarowanej przez użytkownika lub FAŁSZ w przeciwnym razie.

Parametr: wyrażenie, którego wartością jest słowo.

Wynik: wartość logiczna.

Przykłady

```
?pisz określone? "naprzód
FAŁSZ
{NAPRZÓD jest nazwą procedury pierwotnej}
?pisz określone? "procedureczka
FAŁSZ
?oto procedureczka już
PROCEDURECZKA zdefiniowano
?pisz określone? "procedureczka
PRAWDA
?usuń "procedureczka pisz określone? "procedureczka
FAŁSZ
?określ "procedureczka [[][już]]
?pisz określone? "procedureczka
PRAWDA
```

OPISOKNA **DEFINEWINDOW (DFW)**

Składnia: OPISOKNA l

Opis działania: procedura określająca położenie i rozmiar okna graficznego.

Parametr: wyrażenie, którego wartością jest czteroelementowa lista postaci [n n n n], przy czym elementy listy oznaczają kolejno: współrzędną x lewego dolnego rogu okna, współrzędną y lewego dolnego rogu, współrzędną x prawego górnego rogu i współrzędną y prawego górnego rogu.

Uwagi: wymiary okna są zaokrąglane do wielokrotności liczby 8, przy czym współrzędne lewego dolnego rogu są zaokrąglane w dół, a współrzędne prawego górnego rogu w górę. Zatem najmniejsze możliwe okno ma rozmiar 8 * 8 punktów.
Jeżeli wymiary deklarowanego okna przekraczają rozmiary okienka graficzno-tekstowego, to wymiary te są zmniejszane do rozmiarów okienka graficzno-tekstowego bez sygnalizowania błędu.

Przykłady

```
?opisokna [0 0 50 50]
{Zdefiniowanie okna, którego lewy dolny róg leży w początku układu (0,0),
a jego szerokość i wysokość wynosi 56 czyli współrzędne prawego górnego
rogu wynoszą : x=55 i y=55}
?opisokna [1 1 52 50]
{Zdefiniowanie tego samego okna}
?opisokna [0 0 0 0]
{Zdefiniowanie okna o minimalnej szerokości i wysokości równej 8, którego
lewy dolny róg leży w początku układu}
```

OPÓŹNIENIE **DELAY**

Składnia: OPÓŹNIENIE n

Opis działania: procedura zatrzymująca działanie programu na okres n milisekund.

Uwagi: procedura działa podobnie jak procedura CZEKAJ.

OPU

Patrz OPUŚĆ

OPUŚĆ (OPU) **PENDOWN (PD)**

Składnia: OPUŚĆ

Opis działania: procedura "opuść pisak", wprowadzająca pisak żółwia w stan "opuszczony". W tym stanie pisak, w trakcie przesuwania żółwia, zmienia kolor punktów na kolor atramentu.

Uwaga: wykonanie procedury: PODNIEŚĆ (ang. PENUP), ODWRACANIE (ang. PENREVERSE) lub ŚCIERANIE (ang. PENERASE) kończy efekt działania procedury OPUŚĆ. Natomiast wykonanie procedury CZYŚĆ (ang. CLEARSCREEN) i ZMAŻ (ang. CLEAN) nie wpływa na jej działanie.

Przykład

```
?czyść naprzód 20
{Żółw rysuje linię pionową o długości 20 kroków}
?podnieś naprzód 20
{Żółw przesuwa się w pionie o 20 kroków, nie pozostawiając za sobą śladu}
?opuść naprzód 20
{Żółw rysuje linię pionową o długości 20 kroków}
```

ORIENTUJDRUK **SETCOPYORIENTATION (SCO)**

Składnia: ORIENTUJDRUK n

Opis działania: procedura ustala sposb drukowania kopii ekranu lub okna na drukarce: "normalny" (ang. PORTRAIT) lub "odwrócony" (o 90 stopni, ang. LANDSCAPE)

Parametr: 0 orientacja "normalna" (PORTRAIT)
1 orientacja "odwrócona" (LANDSCAPE)

Uwagi: orientacja wydruku nie zależy od typu drukarki.

OST **LAST**

Składnia: OST sl → sl

Opis działania: funkcja "ostatni", której wartością jest ostatni element obiektu będącego parametrem. W przypadku parametru będącego obiektem pustym, wartością funkcji jest obiekt pusty tego samego typu co parametr.

Parametr: wyrażenie, którego wartością jest słowo lub lista.

Wynik: jeśli parametrem jest słowo, to wynikiem jest słowo będące ostatnim znakiem parametru, a jeśli lista, to wynikiem jest jej ostatni element.

Uwaga: podlisty są traktowane jako pojedyncze elementy listy.

Przykłady

```
?pokaż ost [Ala [Ola Jan]]
[ Ola Jan ]
?pokaż ost [[Ala Ola] Jan]
Jan
?pokaż ost ost [Ala [Ola Jan]]
Jan
?pisz ost "Magda
a
?pokaż ost []
[ ]
?pokaż ost "
"
?pisz ost 123
3
?pisz ost 11 + 34
5
?pisz ost slowo "Al "a
a
```

OTO **TO**

Składnia: OTO s parametr_formalny_1 parametr_formalny_2
. . .
wywołanie_procedury
wywołanie_procedury
. . .
wywołanie_procedury
JUŻ

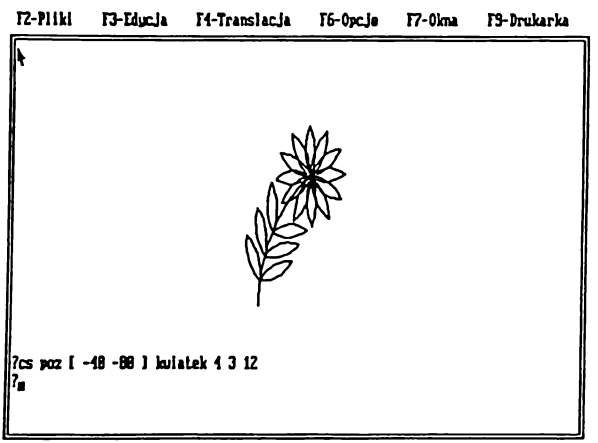
Opis działania: procedura pierwotna rozpoczynająca deklarację procedury. Po OTO występuje nazwa procedury i opcjonalnie nazwy jej parametrów formalnych. Dalej następuje treść procedury, którą tworzą wywołania procedur. Deklarację procedury kończy słowo JUŻ (ang. END).

Uwagi: po OTO, wyjątkowo, słowo będące nazwą deklarowanej procedury nie musi by poprzedzone cudzysłowem.
Procedura OTO może być użyta tylko bezpośrednio w edytorze wierszowym — nie można jej wywoływać z poziomu procedury użytkownika. Nazwa procedury OTO musi ponadto być pierwszym słowem w zredagowanym wierszu edytora wierszowego.

Przykład

```
?oto kwiatek :rozmiar :ile_par_listków :ile_płatków
> powtórz :ile_par_listków
> [
```

```
)    luk :rozmiar / 2 3 / :ile_par_listków
)    lewo 60 platek :rozmiar
)    prawo 90 platek :rozmiar
)    lewo 30
)  ]
) luk :rozmiar * 1.5 3
) powtórz :ile_platków
)  [
)    platek :rozmiar prawo 360 / :ile_platków
)  ]
)już
KWIATEK zdefiniowano
?oto platek :wielkość
) powtórz 2 [ luk :wielkość 6 prawo 120 ]
)już
PŁATEK zdefiniowano
?oto luk :długość :kąt
) powtórz 10
)    [ naprzód :długo  prawo :kąt ]
)już
ŁUK zdefiniowano
?czyść
)?grafika
)?poz [ -40 -80]
)kwiatek 4 3 12
```



Rys. 8.8. Efekt wykonania procedury KWIATEK

PI **PI**

Składnia: PI → n

Opis działania: funkcja, której wartością jest przybliżenie liczby π.

Wynik: liczba równa 3.14159265358979

Przykłady

```
?pisz pi
3.14159265358979
?oto obwód_koła :promień
> wynik 2 * pi * :promień
>już
OBWÓD_KOŁA zdefiniowano
?pisz obwód_koła 10
62.8318530717959
```

PIERW **FIRST**

Składnia: PIERW sl → sl

Opis działania: funkcja "pierwszy", której wartością jest pierwszy element obiektu będącego parametrem. Jeśli parametr jest obiektem pustym, to wartością funkcji jest obiekt pusty tego samego typu co parametr.

Parametr: wyrażenie, którego wartością jest słowo lub lista.

Wynik: jeśli parametrem jest słowo, to wynikiem jest słowo będące pierwszym znakiem parametru; jeśli parametrem jest lista, to wynikiem jest jej pierwszy element.

Uwaga: podlisty są traktowane jako pojedyncze elementy listy.

Przykłady

```
?pokaż pierw [[Ala Ola] Jan]
[ Ala Ola ]
?pokaż pierw [Ala [Ola Jan]]
Ala
?pokaż pierw pierw [[Ala Ola] Jan]
Ala
?pisz pierw pierw [Ala Ola Jan]
A
?pisz pierw "Magda
M
?pokaż pierw []
```

```
[ ]
?pokaż pierw "
"
?pisz pierw 123
1
?pisz pierw 2 + 2 = 4
P
```

PIERWIASTEK (PWK) **SQRT**

Składnia: PIERWIASTEK n → n

Opis działania: funkcja, której wartością jest pierwiastek kwadratowy z parametru.

Parametr: wyrażenie arytmetyczne o wartości nieujemnej.

Wynik: pierwiastek kwadratowy z parametru.

Przykład

```
?pisz pierwiastek 16
4
```

PIERWOTNE? **PRIMITIVEP**

Składnia: PIERWOTNE? s → b

Opis działania: funkcja o wartości logicznej PRAWDA, jeśli jej parametr jest nazwą procedury pierwotnej lub operatorem, lub FAŁSZ w przeciwnym razie.

Parametr: wyrażenie, którego wartością jest słowo.

Wynik: wartość logiczna.

Przykłady

```
?pisz pierwotne? "falsz
PRAWDA
?pisz pierwotne? falsz
PRAWDA
?pisz pierwotne? slowo "fa "lsz
PRAWDA
?pisz pierwotne? 1 > 0
PRAWDA
?pisz pierwotne? "naprzód
PRAWDA
```

```
?pisz pierwotne? "forward
FAŁSZ
?pisz pierwotne? "i
PRAWDA
?pisz pierwotne? "'j
FAŁSZ
?pisz pierwotne? "=
PRAWDA
?pisz pierwotne? "%
FAŁSZ
```

PISAK **PENCOLOR (PC)**

Składnia: PISAK n

Opis działania: procedura ustalająca kolor atramentu pisaka żółwia. Po wykonaniu procedury PISAK kolory ekranu nie ulegają zmianie, zmienia się natomiast kolor żółwia.

Parametr: wyrażenie, którego wartością jest liczba z zakresu od 0 do 15, będąca numerem koloru. Liczby niecałkowite są zaokrąglane. Liczby większe od 15 są niejawnie przekształcane do zakresu od 0 do 15 za pomocą operacji modulo 16.

Przykład

```
?cs pż pisak 3
{Kolory ekranu nie ulegają zmianie, zmienia się kolor żółwia}
?naprzód 30
{Na ekranie pojawia się linia w ustalonym kolorze z żółwiem w tym samym
kolorze na końcu, kolory pozostałej części ekranu pozostają bez zmian}
```

PISZ (PS) **PRINT (PR)**

Składnia: PISZ nsl
lub: (PISZ sl sl ...)

Opis działania: procedura wyświetlająca na ekranie obiekt będący parametrem. Operator dosłowności (cudzysłów) poprzedzający słowo oraz zewnętrzne nawiasy list nie są wyświetlane. Wydruk jest uzupełniany znakiem nowego wiersza.

Parametr: dowolne wyrażenie. W wersji zachłannej — dowolna, większa od dwóch liczba wyrażeń.

Przykłady

```
?pisz 123
123
?pisz 12 + 5 * 2
```

```
22
?pisz 5 * 8 = 2
FAŁSZ
?pisz "Ala
Ala
?pisz [[Magda dziewczynka][Krzyś chłopiec]]
[ Magda dziewczynka ] [ Krzyś chłopiec ]
?(pisz [Magda dziewczynka] [Krzyś chłopiec])
Magda dziewczynka Krzyś chłopiec
?(pisz "a 1 2)
a12
?(pisz "a\ 1 "\ 2)
a 1 2
```

PISZRS232 **SERIALOUT**

Składnia: PISZRS232

Opis działania: procedura przesyłająca do złącza transmisji szeregowej kolejne znaki obiektu będącego parametrem. Operator dosłowności (cudzysłów) poprzedzający słowo oraz zewnętrzne nawiasy list nie są przesyłane. Przesyłane znaki nie są uzupełniane znakiem nowego wiersza.

Parametr: dowolne wyrażenie.

Przykłady

```
?piszrs232 123
{przesłanie do złącza RS-232 trzech znaków: 1, 2 i 3}
?piszrs232 12 + 5 * 2
{przesłanie do złącza RS-232 dwóch znaków: 2 i 2}
?piszrs232 [1]
{przesłanie do złącza RS-232 znaku 1}
```

PKT **DOT**

Składnia: PKT l

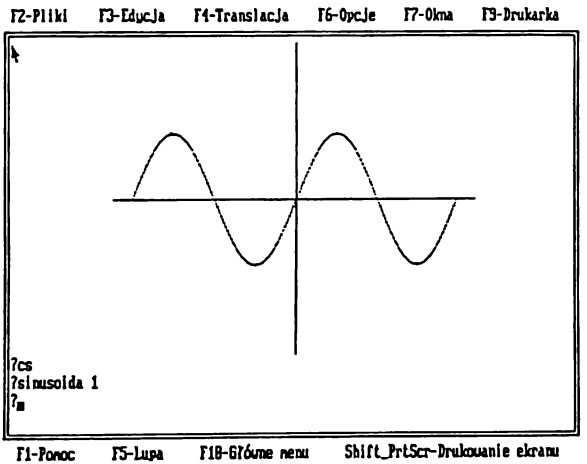
Opis działania: procedura "punkt" rysująca na ekranie punkt o podanych współrzędnych.

Parametr: wyrażenie, którego wartością jest dwuelementowa lista postaci [n n], przy czym pierwszy element oznacza współrzędną lokalną x, a drugi — współrzędną y, oba z zakresu (–32000, 32000).

Przykłady

```
?pkt [20 40]
{Narysowanie na ekranie punktu o współrzędnych x=20 i y=40}
```

```
?oto sinusoida :liczba_okresów
> lokalna "x
>
> układ_współrzędnych
> przypisz "x -360 * :liczba_okresów
> przypisz "liczba_okresów :liczba_okresów * 2
> powtórz 360
> [
>   pkt zdanie :x / :liczba_okresów 50 * sin :x
>   przypisz "x :x + :liczba_okresów
> ]
>już
SINUSOIDA zdefiniowano
oto układ_współrzędnych
> okno sż
> powtórz 4
> [
>   naprzód 200
>   wstecz 200
>   prawo 90
> ]
>już
UKŁAD_WSPÓŁRZĘDNYCH zdefiniowano
?czyść
?sinusoida 1
{Narysowanie wykresu sinusoidy}
```



Rys. 8.9. Rysowanie wykresu sinusoidy za pomocą procedury pierwotnej PKT

PLIKI **FCAT**

Składnia: PLIKI
lub: PLIKI s
w analizie wyrażenia:
PLIKI → l
lub: PLIKI s → l

Opis działania: procedura lub funkcja wyświetlania katalogu dysku w formie rozszerzonej.

Parametr: słowo składające się z (opcjonalnie) nazwy dysku, (opcjonalnie) nazwy katalogu, (opcjonalnie) nazwy pliku (jednoznacznej lub niejednoznacznej). W przypadku braku parametru jest wyświetlany aktualny katalog dysku bieżącego.

Wynik: jeśli funkcja jest wywołana podczas analizy wyrażenia, to wynikiem jej działania jest lista zawierająca słowa odpowiadające nazwom wszystkich wybranych plików.

Przykłady

```
?pliki "*.log
Katalog C:\LOGO\PLIKI
ALA.LOG      1034B 12/05/1992 12:00
OLA.LOG       189B 11/05/1992 18:21
?pisz pliki "*.log
ALA.LOG OLA.LOG
```

PŁOT **FENCE**

Składnia: PŁOT

Opis działania: procedura przejścia do takiego trybu okna graficznego, w którym wyjście żółwia poza zadeklarowane okienko graficzne jest traktowane jako błąd. Efekt działania procedury kończy się po wykonaniu procedury OKNO (ang. WINDOW) lub SKLEJ (ang. WRAP).

Przykład

```
?plot
?prawo 10 naprzód 1000
Żółw poza oknem
```

PM **SHOWMOUSE (SM)**

Składnia: PM

Opis działania: procedura "pokaż kursor myszki", powodująca przejście do trybu rysowania, w którym kursor myszki podczas wykonywania procedur LOGO jest widoczny.

Przykład

```
?czyść sm czekaj 50
{Kursor myszki przestaje być widoczny na czas jednej sekundy}
?pm
{Kursor myszki staje się widoczny}
```

PO **PRINTOUT (PO)**

Składnia: PO s

Opis działania: procedura "pokaż treść lub wartość", wyświetlająca treść procedury lub wartość zmiennej, której nazwa jest parametrem.

Parametr: wyrażenie, którego wartością jest słowo będące nazwą procedury lub zmiennej.

Uwagi: treść procedury zadeklarowanej procedurą OKREŚL (ang. DEFINE) nie może być wyświetlana procedurą PO.
Jeżeli w programie istnieje zmienna i procedura o tej samej nazwie, to zostanie wyświetlony tekst procedury oraz wartość zmiennej.

Przykład

```
?oto przysłowie
> wynik [baba z wozu koniom lżej]
>już
PRZYSŁOWIE zdefiniowano
?przypisz "przysłowie "powiedzonko
?po "przysłowie
oto przysłowie
  wynik [baba z wozu koniom lżej]
już

PRZYPISZ "PRZYSŁOWIE "powiedzonko
```

POD

Patrz PODNIEŚ

PODNIEŚ (POD) **PENUP (PU)**

Składnia: PODNIEŚ

Opis działania: procedura "podnieś pisak" zmieniająca stan pisaka żółwia tak, aby żółw poruszając się nie zostawiał za sobą śladu.

Uwaga: wykonanie procedury OPUŚĆ (ang. PENDOWN), ŚCIERANIE (ang. PENERASE) i ODWRACANIE (ang. PENREVERSE) kończy efekty działania procedury PODNIEŚ. Natomiast wykonanie procedury CZYŚĆ (ang. CLEARSCREEN) i ZMAŻ (ang. CLEAN) nie wpływa na jej działanie.

Przykład

```
?podnieś naprzód 20
{Żółw przesuwa się w pionie o 20 kroków, nie pozostawiając za sobą śladu}
?opuść naprzód 20
{Żółw rysuje linię pionową o długości 20 kroków}
```

POKAŻ **SHOW**

Składnia: POKAŻ sl

Opis działania: procedura wyświetlająca na ekranie obiekt będący parametrem i znak nowego wiersza. Listy są wyświetlane wraz z nawiasami, a słowa łącznie z cudzysłowem.

Parametr: dowolne wyrażenie.

Przykład

```
?pokaż 123
123
?pokaż "123
"123
?pokaż "Ala
"Ala
?pokaż [Magda Krzyś]
[ Magda Krzyś ]
?pokaż "a < "b
PRAWDA
```

PONP **POTS**

Składnia: PONP

Opis działania: procedura "pokaż wszystkie nagłówki procedur", wyświetlająca nazwy wszystkich aktualnie zadeklarowanych procedur wraz z nazwami ich parametrów formalnych.

Przykład

```
?oto przysłowie
> wynik [najlepsza jest baba pieczona]
>już
PRZYSŁOWIE zdefiniowano
?określ "sinus [[:x][(pisz "'sin(' :x ")=' sin :x) już]]
?ponp
OTO przysłowie
OTO "sinus :x
```

POW **POALL**

Składnia: POW

Opis działania: procedura "pokaż wszystko", wyświetlająca treść wszystkich procedur zadeklarowanych procedurą pierwotną OTO (ang. TO) oraz wszystkich nazw i wartości zmiennych nie przesłoniętych w miejscu wywołania procedury POW.

Uwaga: treść procedur zadeklarowanych procedurą pierwotną OKREŚL (ang. DEFINE) nie jest wyświetlana.

Przykład

```
?oto przysłowie
> wynik [Nie święci garnki lepią]
>już
PRZYSŁOWIE zdefiniowano
?określ "garncarz [[:kto][wynik zdanie :kto [garnki lepi] już]]
?przyp "a [Ala ma kota]
?pow
oto przysłowie
 wynjk [Nie święci garnki lepią]
już
PRZYPISZ "A [ Ala ma kota ]
```

POWP **POPS**

Składnia: POWP

Opis działania: procedura "pokaż wszystkie procedury", wyświetlająca treść wszystkich procedur zadeklarowanych za pomocą procedury OTO (ang. TO).

Uwaga: treść procedur zadeklarowanych procedurą pierwotną OKREŚL (ang. DEFINE) nie jest wyświetlana.

Przykład

```
?oto przysłowie
> wynik [żona gruba — męża chluba]
>już
PRZYSŁOWIE zdefiniowano
?powp
oto przysłowie
 wynik [żona gruba — męża chluba]
już
```

POWTÓRZ — REPEAT

Składnia: POWTÓRZ n l

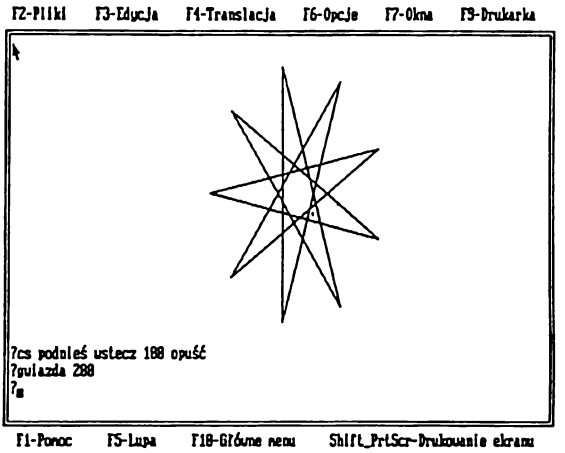
Opis działania: procedura powodująca wielokrotne powtórzenie wykonania listy procedur stanowiącej drugi parametr. Liczba powtórzeń jest określona przez pierwszy parametr.

Parametry: pierwszym parametrem jest wyrażenie, którego wartością jest liczba powtórzeń z zakresu (0, 65535); drugim parametrem jest wyrażenie, którego wartością jest lista wywołań procedur wraz z parametrami aktualnymi.

Uwagi: liczba powtórzeń jest zaokrąglana do liczby całkowitej. Jeżeli liczba powtórzeń jest równa zeru, to lista będąca drugim parametrem nie jest wykonywana.

Przykłady

```
?powtórz 5 [pisz "Ala]
Ala
Ala
Ala
Ala
Ala
?oto gwiazda :bok
> powtórz 9
> [ naprzód :bok prawo 160 ]
>już
 GWIAZDA zdefiniowano
?wstecz 100
?gwiazda 200
{Rysowanie gwiazdy dziewięcioramiennej}
```



Rys. 8.10. Przykład zastosowania procedury pierwotnej POWTÓRZ

POWZ **PONS**

Składnia: POWZ

Opis działania: procedura "pokaż wszystkie zmienne", wyświetlająca nazwy i wartości wszystkich zmiennych nie przesłoniętych w miejscu wywołania tej procedury.

Przykład:

```
?oto a :a
>powz
>już
A zdefiniowano
?przyp "a [Ala ma kota]
?powz
PRZYPISZ "A [ Ala ma kota ]
?a 1
PRZYPISZ "A 1
?powz
PRZYPISZ "A [ Ala ma kota ]
```

POZ **SETXY**

Składnia: POZ 1

Opis działania: procedura przenosząca żółwia do punktu o współrzędnych

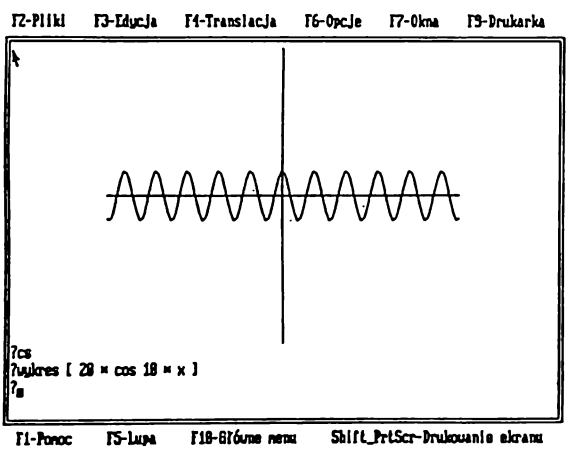
x i y będących parametrem. Ustawienie żółwia nie ulega zmianie.

Parametr: wyrażenie, którego wartością jest dwuelementowa lista postaci [n n], przy czym pierwszy element oznacza współrzędną x, a drugi — współrzędną y, oba z zakresu (–32000, 32000).

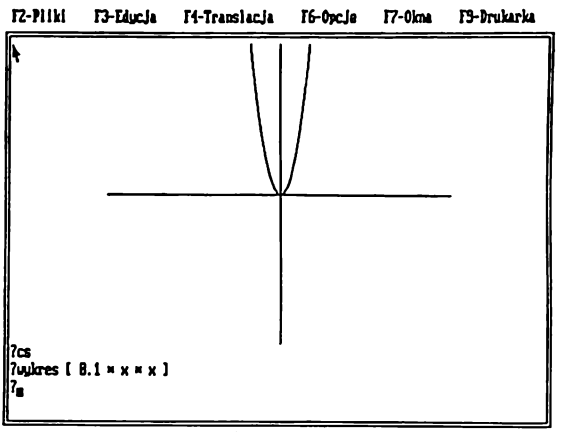
Przykład

```
?oto wykres :funkcja
>{parametr jest nazwą funkcji lub listą zawierającą wyrażenie;
> przekazanie parametru jest przekazaniem "przez procedurę"}
> sż okno czyść
> układ_współrzędnych
> przypisz "x –200
> podnieś  poz zdanie :x zrób :funkcja  opuść
> powtórz 200
>  [ przypisz "x :x + 2   poz zdanie :x zrób :funkcja ]
>już
WYKRES zdefiniowano
?oto układ_współrzędnych
> powtórz 4 [ naprzód 200 wstecz 200 prawo 90 ]
>już
UKŁAD_WSPÓŁRZĘDNYCH zdefiniowano
?oto x wynik :x już
X zdefiniowano
?wykres [20 * cos 10 * x]
?wykres [0.1 * x * x]
```

a)



b)



Rys. 8.11. Przykłady zastosowania procedury pierwotnej POZ:
a) wykres funkcji cosinus;
b) wykres funkcji kwadratowej

POZ? **POSITION (POS)**

Składnia: POZ? → l

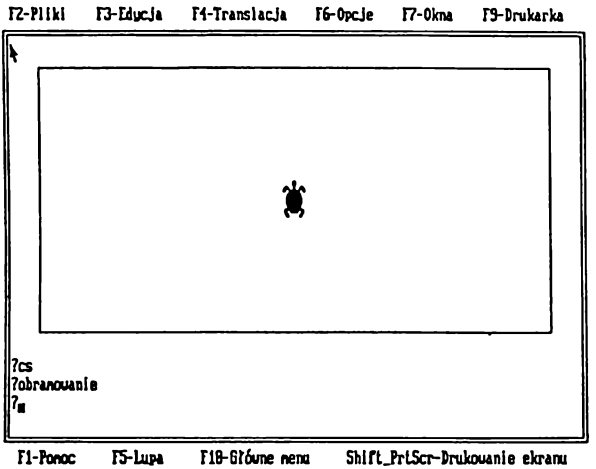
Opis działania: funkcja, której wartością jest dwuelementowa lista określająca aktualne współrzędne żółwia.

Wynik: dwuelementowa lista postaci [n n], przy czym pierwszy element oznacza współrzędną x, a drugi — współrzędną y żółwia.

Przykłady

```
?czyść naprzód 10
?pisz poz?
 0 10
?oto obramowanie
> lokalna "pozycja
> przypisz "pozycja poz?
> podnieś sż
> poz [280 100]
> opuść
> pozy -100
> pozx -280
> pozy 100
```

```
> pozx 280
> podnieś poz :pozycja
> pż opuść
>już
OBRAMOWANIE zdefiniowano
?obramowanie
{Otoczenie standardowego okna graficznego ramką bez zmiany pozycji żółwia}
```



Rys. 8.12. Efekt wykonania procedury OBRAMOWANIE

POZMYSZKI? **MOUSEPOSITION (MPOS)**

Składnia: POZMYSZKI? → l

Opis działania: funkcja, której wartością jest dwuelementowa lista określająca aktualne współrzędne myszki.

Wynik: dwuelementowa lista postaci [n n], przy czym pierwszy element oznacza współrzędną x, a drugi — współrzędną y myszki; współrzędne myszki są liczone tak samo, jak współrzędne żółwia.

Przykłady

```
?cs pisz pozmyszki?
-300 100
{przesunięcie myszki w okolice środka okienka graficznego}
?pisz pozmyszki?
10 12
```

POZX **SETX**

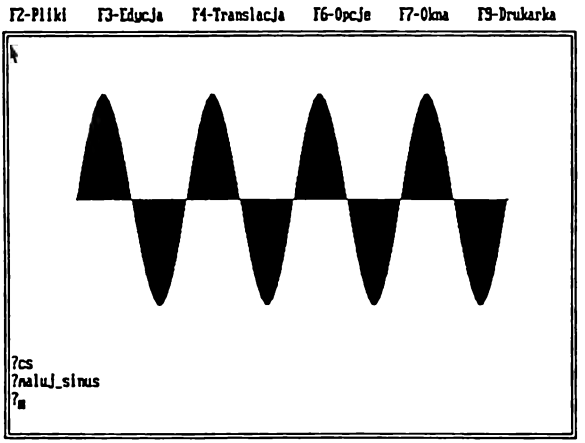
Składnia: POZX n

Opis działania: procedura przesuwająca żółwia w poziomie do punktu o współrzędnej x będącej parametrem. Kierunek ustawienia żółwia nie ulega zmianie.

Parametr: wyrażenie arytmetyczne o wartości z zakresu (–32000, 32000), określające współrzędną x.

Przykład

```
?oto maluj_sinus
› sż czyść opuść
› przyp "x –240 pozx :x
› powtórz 481
›   [ pozy 80 * sin :x * 3 pozy 0 przyp "x :x + 1 pozx :x ]
›już
MALUJ_SINUS zdefiniowano
?maluj_sinus
```



Rys. 8.13. Efekt wykonania procedury MALUJ_SINUS

POZX? **XCOR**

Składnia: POZX? → n

Opis działania: funkcja, której wartością jest liczba równa współrzędnej x żółwia.

Wynik: liczba.

Przykład

```
?wróć pisz pozx?
0
```

POZY **SETY**

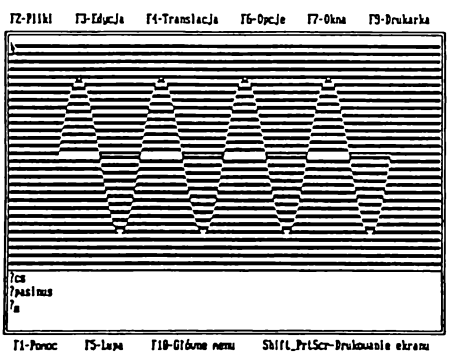
Składnia: POZY n

Opis działania: procedura przesuwająca żółwia w pionie do punktu o współrzędnej y będącej parametrem. Kierunek ustawienia żółwia nie ulega zmianie.

Parametr: wyrażenie arytmetyczne o wartości z zakresu (–32000, 32000), określające współrzędną y.

Przykład

```
?oto pasinus
> wzór [#ff #ff #ff #ff 0 0 0 0]
> sż podnieś czyść
> zamaluj
> przyp "x -240
> pozx :x
> powtórz 481
>  [ odwracanie pozy 80 * sin :x * 3 podnieś pozy 0
>    przyp "x :x + 1 pozx : ]
>już
PASINUS zdefiniowano
?pasinus
```



Rys. 8.14. Efekt wykonania procedury PASINUS

POZY? **YCOR**

Składnia: POZY? → n

Opis działania: funkcja, której wartością jest liczba równa współrzędnej y żółwia.

Wynik: liczba.

Przykład

```
?wróć pisz pozy?
0
```

PPM

Patrz PRAWYPRZYCISK

PRAWDA **TRUE**

Składnia: PRAWDA → b

Opis działania: funkcja, której wartością jest stała logiczna PRAWDA.

PRAWO (PW) **RIGHT (RT)**

Składnia: PRAWO n

Opis działania: procedura obracająca żółwia w prawo.

Parametr: wyrażenie, którego wartością jest kąt obrotu w stopniach.

Uwaga: dla ujemnej wartości kąta następuje obrót w lewo.

Przykład

```
?czyść
?prawo 410
?pisz kąt?
50
```

PRAWYPRZYCISK (PPM) **MOUSERIGHTBUTTON (MRB)**

Składnia: PRAWYPRZYCISK → b

Opis działania: funkcja o wartości logicznej PRAWDA, jeśli w momencie wywołania procedury był naciśnięty prawy przycisk myszki.

Wynik: wartość logiczna.

Uwaga: poprzednie naciśnięcia klawiszy myszki nie są pamiętane (w odróżnieniu od funkcji KLAWISZ?).

Przykład

```
?powtórz 10 [jeśli prawyprzycisk [ps "tak] inaczej [ps "nie]]
nie
nie
nie
nie
nie
tak          {przyciśnięty prawy przycisk myszki}
tak
tak
nie          {prawy przycisk zwolniony}
nie
```

PROPORCJA **SETSCRUNCH**

Składnia: PROPORCJA l

Opis działania: procedura ustalająca jednostki na osiach układu współrzędnych. Standardowo jednostki te wynoszą 1 punkt na obu osiach.

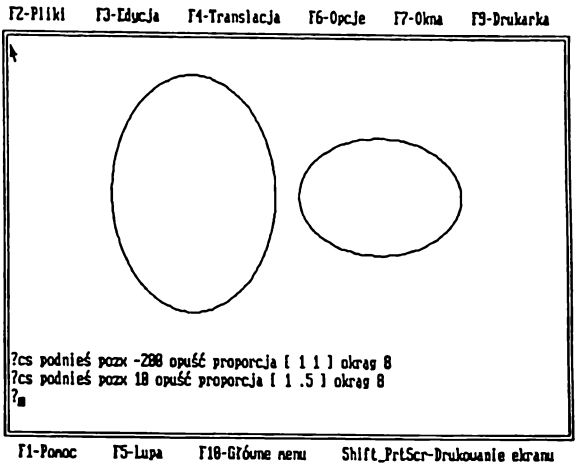
Parametr: wyrażenie, którego wartością jest dwuelementowa lista postaci [n n], przy czym pierwszy element oznacza jednostkę na osi x, a drugi — jednostkę na osi y. Jednostki są wyrażane w liczbie punktów ekranu.

Uwaga: jeżeli jednostka x jest różna od jednostki y, to ustawienie żółwia nie odpowiada w pełni kierunkowi jego poruszania.

Przykład

```
?czyść proporcja [1 2] prawo 45 naprzód 40
?proporcja [1 1]
?oto okrąg :rozmiar
> powtórz 72
>   [
>     naprzód :rozmiar
>     prawo 5
>   ]
>już
OKRĄG zdefiniowano
?czyść
?sż
```

```
?okrąg 8
{Narysowanie okręgu}
?proporcja [1 .5]
?okrąg 8
{Narysowanie elipsy}
```



Rys. 8.15. Przykład zastosowania procedury pierwotnej PROPORCJA

PROPORCJA? **SCRUNCH**

Składnia: PROPORCJA? → l

Opis działania: funkcja, której wartością jest dwuelementowa lista zawierająca aktualne jednostki na osiach układu współrzędnych. Standardowo PROPORCJA? = [1 1].

Wynik: dwuelementowa lista postaci [n n], przy czym pierwszy element oznacza jednostkę na osi x, a drugi — jednostkę na osi y. Jednostki są wyrażane w liczbie punktów ekranu.

Przykłady

```
?pisz proporcja?
 1 1
?proporcja [2 1]
?pisz proporcja?
 2 1
```

```
?oto równe_proporcje?
> wynik równe? pierw proporcja? ost proporcja?
>już
RÓWNE_PROPORCJE? zdefiniowano
?pisz równe_proporcje?
FAŁSZ
?proporcja [2 2]
?pisz równe_proporcje?
PRAWDA
```

PRZERWIJ **TOPLEVEL**

Składnia: PRZERWIJ

Opis działania: przerwanie wykonywania programu i przejście do interpretatora poleceń LOGO.

Przykład

```
?oto logarytm :x
> jeśli nie liczba? :x
> [
>   wpisz [Błąd — parametr nie jest liczbą !]
>   przerwij
> ]
> jeśli :x <= 0
> [
>   wpisz [Błąd — parametr nie jest liczbą dodatnią !]
>   przerwij
> ]
> wynik ln :x
>już
LOGARYTM zdefiniowano
?pisz logarytm []
 Błąd — parametr nie jest liczbą !
*** Przerwanie *** na poziomie 1 w wierszu 4 w procedurze LOGARYTM
?pisz logarytm 0
 Błąd — parametr nie jest liczbą dodatnią !
*** Przerwanie *** na poziomie 1 w wierszu 9 w procedurze LOGARYTM
?pisz logarytm 1
0
```

PRZYP

Patrz PRZYPISZ

PRZYPISZ (PRZYP) **MAKE**

Składnia: PRZYPISZ s sl

Opis działania: procedura nadająca zmiennej o nazwie będącej pierwszym parametrem wartość będącą drugim parametrem.

Parametry: pierwszy parametr jest wyrażeniem, którego wartością jest słowo określające nazwę zmiennej; drugi parametr jest dowolnym wyrażeniem.

Uwagi: wartością zmiennej może być w szczególności nazwa innej zmiennej. Dostęp do wartości zmiennej uzyskuje się za pomocą funkcji WARTOŚĆ (ang. THING) lub za pomocą operatora wartościowania — dwukropka.

Przykłady

```
?przypisz "wzrost 100 + 75
?jeśli :wzrost > 140 [w kapeluszu] [pisz [wzrost wysoki]]
wzrost wysoki
?przypisz "lista_osób [[kobieta Joanna][mężczyzna Adam]]
?oto komplement :kto
> jeśli pierw: kto = "kobieta
>    [pisz zdanie ost :kto [jest piękna]]
> inaczej
>    [pisz zdanie ost :kto [jest przystojny]]
>już
KOMPLEMENT zdefiniowano
?komplement pierw :lista_osób
Joanna jest piękna
?komplement element 2 :lista_osób
Adam jest przystojny
?przypisz "dziewczynka "Ala
?pisz :dziewczynka
Ala
?przypisz "dziecko "dziewczynka
?pisz :dziecko
dziewczynka
?pisz wartość "dziecko
dziewczynka
?pisz wartość wartość "dziecko
Ala
```

PS

Patrz PISZ

PUNKT? **POINTP**

Składnia: PUNKT? l → b

Opis działania: funkcja, której wartością logiczną jest PRAWDA jeśli punkt o współrzędnych wskazywanych przez listę ma kolor pisaka lub FAŁSZ w przeciwnym razie.

Parametr: dwuelementowa lista postaci [n n], przy czym pierwszy element oznacza współrzędną x, a drugi współrzędną y punktu.

Wynik: wartość logiczna.

Przykłady

```
?cs
?ps punkt? [ 0 0 ]
FAŁSZ
?naprzód 1
?ps punkt? [ 0 0 ]
PRAWDA
```

PUSTE? **EMPTYP**

Składnia: PUSTE? sl → b

Opis działania: funkcja, której wartością logiczną jest PRAWDA jeśli obiekt będący jej parametrem jest pusty lub FAŁSZ w przeciwnym razie.

Parametr: wyrażenie, którego wartością jest słowo lub lista.

Wynik: wartość logiczna.

Przykłady

```
?pisz puste? []
PRAWDA
?pisz puste? "A
FAŁSZ
?pisz puste? [12]
FAŁSZ
?pisz puste? [[]]
FAŁSZ
?pisz puste? "
PRAWDA
?pisz puste? ""
FAŁSZ
```

```
?oto ma_polskie_znaki? :slowo
> wynik ma_znaki? ml :slowo [ ą ę ć ł ń ó ś ż ź ]
>już
MA_POLSKIE_ZNAKI? zdefiniowano
?oto ma_znaki? :slowo :lista
>{czy slowo ma znaki występujące na liście ?}
> jeśli puste? :lista
>    [ wynik falsz ]
> jeśli jest_znak? pierw :lista :slowo
>    [ wynik prawda ]
> wynik ma_znaki? :slowo bezpierw :lista
>już
MA_ZNAKI? zdefiniowano
?oto jest_znak? :znak :slowo
>{czy znak występuje w slowie ?}
> jeśli puste? :slowo
>    [ wynik falsz ]
> jeśli :znak = pierw :slowo
>    [ wynik prawda ]
> wynik jest_znak? :znak bezpierw :slowo
>już
JEST_ZNAK? zdefiniowano
?pisz ma_polskie_znaki? "język
PRAWDA
```

PW

Patrz PRAWO

PWK

Patrz PIERWIASTEK

PŻ **SHOWTURTLE (ST)**

Składnia: PŻ

Opis działania: procedura "pokaż żółwia", powodująca przejście do trybu rysowania, w którym żółw jest widoczny.

Przykład

```
?czyść sż
{Żółw przestaje być widoczny}
?pż
{Żółw staje się widoczny}
```

RAMKA **BORDERCOLOR (BC)**

Składnia: RAMKA n

Opis działania: procedura ustalająca kolor ramki ekranu.

Parametr: wyrażenie, którego wartością jest liczba z zakresu od 0 do 15, będąca numerem koloru. Liczby niecałkowite są zaokrąglane. Liczby większe od 15 są niejawnie przekształcane do zakresu od 0 do 15 za pomocą operacji modulo 16.

Uwagi: dla karty graficznej HGC i większości kart EGA/VGA nie jest możliwe wybranie innego koloru ramki niż czarny.

RED **EDIT (ED)**

Składnia: RED sl
lub: RED

Opis działania: procedura "redaguj" powodująca przejście do edytora ekranowego.

Parametr: słowo lub lista postaci [s s ...]. Parametr będący słowem jest interpretowany jako nazwa redagowanej procedury. Podobnie słowa będące elementami listy, która jest parametrem, są interpretowane jako nazwy redagowanych procedur. Bezparametrowe wywołanie procedury pierwotnej RED powoduje przejście do edytora ekranowego i wyświetlenie tekstu ostatnio redagowanych procedur.

Uwagi: parametr musi być pojedynczym słowem lub listą. Parametr będący słowem lub listą pustą oznacza wywołanie edytora ekranowego z pustym obszarem roboczym w celu redakcji nowych procedur. Dokładny opis możliwości edytora ekranowego zawarto w rozdziale 6.

Przykłady

```
?red "kwadrat
{Przejście do redakcji procedury KWADRAT}
?red [koło kwadrat]
{przejście do redakcji procedur KOŁO oraz KWADRAT}
?red []
{Przejście do redakcji nowych procedur}
?oto poprawki
> red
> kwadrat
> czekaj_na_klawisz
> poprawki
>już
```

```
POPRAWKI zdefiniowano
?oto czekaj_na_klawisz
> jeśli klawisz? [pisz czytajznak stop]
> czekaj_na_klawisz
>już
CZEKAJ_NA_KLAWISZ zdefiniowano
?oto kwadrat
> powtórz 4
>   [np 40 pw 90]
>już
KWADRAT
?poprawki
```
{Rozpoczęcie cyklicznej redakcji i wykonywania procedury KWADRAT. Cykl może być przerwany w edytorze przez zaniechanie edycji (^KQ) lub w czasie wykonywania procedury przez wciśnięcie na klawiaturze klawiszy Ctrl–Break.}

REDW **EDALL**

Składnia: REDW

Opis działania: procedura "redaguj wszystko", wywołująca edytor w celu redagowania wszystkich procedur zadeklarowanych uprzednio procedurą OTO (ang. TO). Wywołanie procedury REDW nie dotyczy procedur zadeklarowanych procedurą OKREŚL (ang. DEFINE).

Uwagi: procedura REDW umożliwia także redagowanie nowych procedur. Dokładny opis możliwości edytora ekranowego znajduje się w rozdziale 6.

Przykład

```
?redw
```
{Na ekranie pojawia się okno edytora ekranowego; powrót do edytora wierszowego następuje po wydaniu polecenia przerwania edycji ^KQ lub zakończenia edycji i zdefiniowania redagowanych procedur ^KD. W dowolnym momencie można powrócić do redagowania tekstu przez naciśnięcie klawisza F3 lub wywołanie procedury pierwotnej ED bez parametrów.}

RESZTA **REMAINDER**

Składnia: RESZTA n n → n

Opis działania: funkcja, której wartością jest reszta z dzielenia całkowitego pierwszego parametru przez drugi.

Parametry: dwa wyrażenia arytmetyczne.

Wynik: liczba.

Przykłady

```
?pisz reszta 5 3
2
?pisz reszta 7.3 2
1.3
?pisz reszta 7.5 2.1
1.2
?pisz reszta -7 -2
1
```

RÓWNE? **EQUALP**

Składnia: RÓWNE sl sl → b

Opis działania: funkcja, której wartością jest PRAWDA, jeśli oba obiekty będące jej parametrami są równe, czyli są tego samego typu i mają takie same elementy, w tej samej kolejności. Jeśli obiekty będące parametrami funkcji nie są równe, to wartością funkcji jest FAŁSZ.

Parametry: dwa dowolne wyrażenia.

Wynik: wartość logiczna.

Przykłady

```
?pisz równe? 5 5
PRAWDA
?pisz równe? 5 "5
PRAWDA
?pisz równe? 5 [5]
FAŁSZ
?pisz równe? 5 pierw [5]
PRAWDA
?pisz równe? Ala "Ala
Nie wiem jak zrobić ALA
{Ala — wywołanie niezadeklarowanej procedury}
oto "zero
> wynik 0
>już
ZERO zdefiniowano
?pisz równe? 0 zero
PRAWDA
?pisz równe? [Ala] "Ala
FAŁSZ
?pisz równe? pierw ["Ala] "Ala
PRAWDA
```

```
?pisz równe? pierw [Ala] "Ala
PRAWDA
?pisz równe? [Ala Ola Jan] [Ala Ola [ Jan ]]
FAŁSZ
?pisz równe? [Ala Ola Jan] [Ala Ola Jan]
PRAWDA
?pisz równe? falsz falsz
PRAWDA
?pisz równe? falsz [falsz]
FAŁSZ
?pisz równe? falsz 2 + 2 = 5
PRAWDA
```

SIGN **SIGN**

Składnia: SIGN n → n

Opis działania: funkcja signum podająca znak parametru.

Parametr: wyrażenie arytmetyczne.

Wynik: liczba o wartości 1 dla parametru dodatniego, 0 dla parametru równego zeru i –1 dla parametru ujemnego.

Przykłady

```
?pisz sign 17
1
?pisz sign 0
0
?pisz sign -56.25 + 3.87
-1
```

SIN **SIN**

Składnia: SIN n → n

Opis działania: funkcja trygonometryczna sinus.

Parametr: wyrażenie, którego wartością jest kąt w stopniach.

Wynik: liczba równa sinusowi parametru.

Przykłady

```
?pisz sin 0
0
?pisz sin 30
0.5
```

SKALUJDRUK **SETCOPYSCALE (SCS)**

Składnia: SKALUJDRUK l

Opis działania: procedura ustalająca rozmiar wydruku na drukarce dla kopii ekranu i okna graficznego.

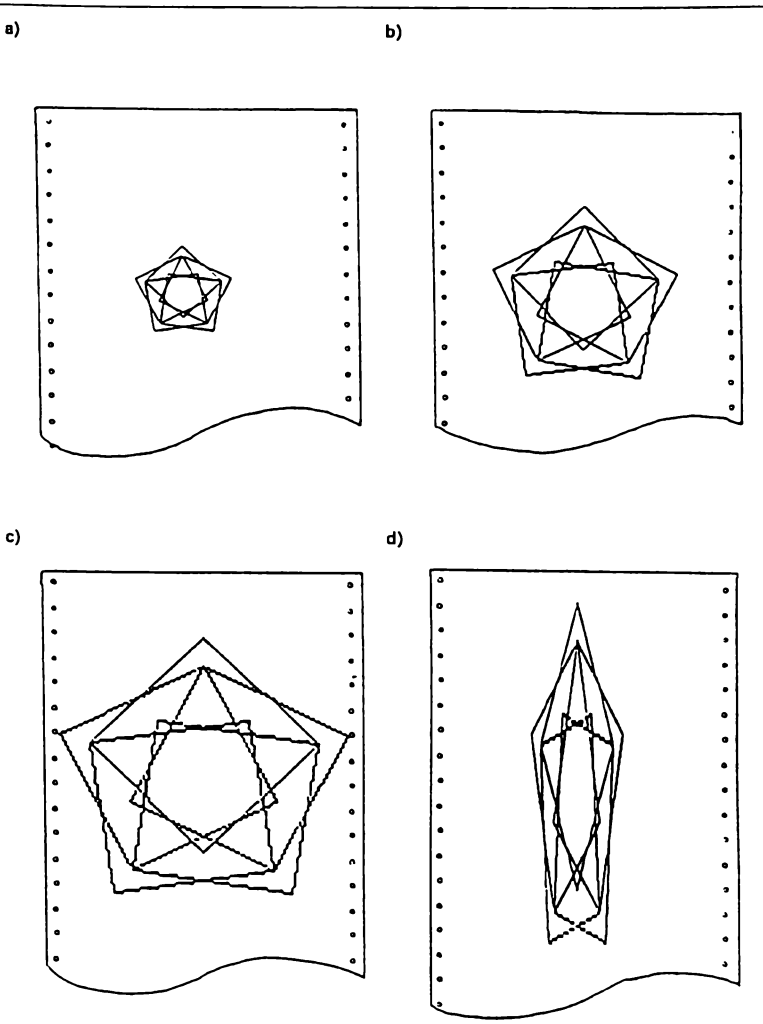
Parametr: wyrażenie, którego wartością jest dwuelementowa lista postaci [n n]. Jej elementy są liczbami z zakresu (1, 99) ustalające odpowiednio skalę wydruku w osi X i Y. Wartość skali można interpretować jako liczbę igieł użytych w danym kierunku (poziomym lub pionowym) do wydrukowania jednego punktu ekranu. Liczby niecałkowite są zaokrąglane. Po wywołaniu systemu programowania AC–LOGO skala wydruku jest ustalana na podstawie parametru PE pliku konfiguracyjnego (por. Dodatek F).

Uwagi: procedura przeskalowuje wydruk w przypadku wykonywania kopii ekranu za pomocą procedury DRUKUJOBRAZ (ang. COPYSCREEN) oraz okna graficznego za pomocą procedury DRUKUJOKNO (ang. COPYWINDOW). Nie wpływa natomiast na skalę wydruku w przypadku tekstu drukowanego jako echo po wykonaniu procedury WŁĄCZDRUK (ang. PRINTON).

Użycie dużej skali wydruku tzn. żądanie wydruku na drukarce większej liczby punktów w osi X niż pozwala na to szerokość wałka drukarki, ma konsekwencje zależne od typu drukarki. Najczęściej "wystająca" część wydruku jest pomijana. Skalę wydruku w osi X można dodatkowo zmieniać za pomocą zmiany trybu pracy drukarki (por. procedura TRYBDRUKU, ang. SETCOPYMODE).

Przykład

```
?oto figura
> podnieś pozx –40 opuść prawo 9
> powtórz 8 [ powtórz 6 [ naprzód 6 prawo 90 ] prawo 36 ]
>już
FIGURA zdefiniowano
?czyść sż figura
?skalujdruk [1 1] drukujokno skalujdruk [2 2] drukujokno
?skalujdruk [3 3] drukujokno skalujdruk [1 4] drukujokno
```



Rys. 8.16. Wydruki okna graficznego po wykonaniu procedury:
a) SKALUJDRUK [1 1]; b) SKALUJDRUK [2 2];
c) SKALUJDRUK [3 3]; d) SKALUJDRUK [1 4]

SKLEJ **WRAP**

Składnia: SKLEJ

Opis działania: procedura powoduje przejście do standardowego trybu okna graficznego, w którym po przekroczeniu krawędzi okna żółw ukazuje się po jego przeciwległej stronie. Efekt działania procedury kończy się po wykonaniu procedury OKNO (ang. WINDOW) lub PŁOT (ang. FENCE).

Przykład

```
?sklej
?prawo 10 naprzód 1000
```

SŁOWO **WORD**

Składnia: SŁOWO s s → s
lub: (SŁOWO s s ...) → s

Opis działania: funkcja, której wartością jest słowo utworzone przez połączenie słów będących jej parametrami.

Parametry: dwa wyrażenia, których wartościami są słowa. W wersji zachłannej — dowolna, większa od jednego liczba wyrażeń, których wartościami są słowa.

Wynik: słowo będące połączeniem parametrów.

Przykłady

```
?pisz słowo "Ma "gda
Magda
?(pisz słowo "A "l "i "n "a)
Alina
?zrób zdanie słowo "pi "sz 1
1
?pisz równe? "lwona (słowo "l "wo "na)
PRAWDA
```

SŁOWO? **WORDP**

Składnia: SŁOWO sl → b

Opis działania: funkcja o wartości logicznej PRAWDA, jeśli obiekt będący jej parametrem jest słowem.

Parametr: dowolne wyrażenie.

Wynik: wartość logiczna.

Przykład

```
?pisz slowo? "alfa
PRAWDA
?pisz slowo? 123
PRAWDA
?pisz slowo? "123
PRAWDA
?pisz slowo? 2 > 1
PRAWDA
?oto ala
>wynik "ala
już
ALA zdefiniowano
?pisz slowo? ala
PRAWDA
?oto ola
>wynik [ola]
już
OLA zdefiniowano
?pisz slowo? ola
FAŁSZ
?pisz slowo [ Ala Ola ]
FAŁSZ
```

SM **HIDEMOUSE (HM)**

Składnia: SM

Opis działania: procedura "schowaj kursor myszki", powodująca przejście do takiego trybu rysowania, w którym kursor myszki nie jest widoczny podczas wykonywania procedur LOGO. Tryb ten jest użyteczny, gdy sylwetka kursora myszki zaciemnia rysunek lub gdy jest celowe zwiększenie szybkości rysowania.

Uwaga: kursor myszki można schować tylko na czas wykonywania procedur LOGO. W czasie wykonywania poleceń systemu okienkowego lub redagowania za pomocą edytora kursor ten jest zawsze widoczny.
Kursor myszki można schować także za pomocą polecenia z pliku konfiguracyjnego (por. dodatek F).

SORTUJ **SORT**

Składnia: SORTUJ l → l
Opis działania: funkcja, której wartością jest posortowana lista słów będąca jej parametrem. Sortowanie polega na porównywaniu kolejnych słów zgodnie z zasadami opisanymi w rozdziale 3.

Parametr: wyrażenie, którego wartością jest lista zawierająca słowa lub lista pusta.

Wynik: lista zawierająca słowa lub lista pusta.

Przykłady

```
?pokaż sortuj []
[ ]
?pisz sortuj [Ala ma kota. To jest kot Ali.]
Ala Ali. To jest kot kota. ma
```

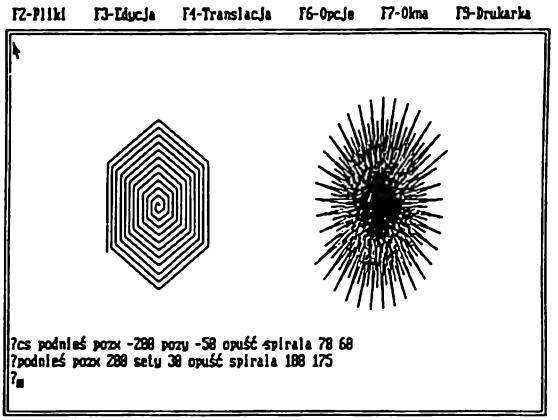
STOP **STOP**

Składnia: STOP

Opis działania: procedura powodująca zakończenie działania aktualnie wykonywanej procedury i przejście do procedury, z której ją wywołano. Wywołanie procedury STOP poza tekstem procedury użytkownika (w edytorze wierszowym) powoduje powtórne wywołanie edytora wierszowego (przerwanie wykonywania programu).

Przykład

```
?oto spirala :bok :kąt
> jeśli :bok <= 0 [stop]
> naprzód :bok prawo :kąt
> spirala :bok — 1 :kąt
>już
SPIRALA zdefiniowano
?sż czyść spirala 70 60 wstecz 40 spirala 180 175
```



Rys. 8.17. Rysunki wykonane za pomocą procedury SPIRALA

STOPER **TIMER**

Składnia: STOPER → n

Opis działania: funkcja podająca liczbę sekund i setnych sekundy, jakie upłynęły od czasu ostatniego wykonania procedury ODLICZANIE (ang. STARTTIMER) lub od momentu uruchomienia programu LOGO.

Wynik: liczba większa lub równa zero.

Przykłady

```
?ps stoper
123.45
{czas, jaki upłynął od momentu uruchomienia programu}
?odliczanie ps stoper
0
```

SUMA **SUM**

Składnia: SUMA n n → n
lub: (SUMA n n ...) → n

Opis działania: funkcja, której wartością jest suma parametrów.

Parametry: dwa wyrażenia arytmetyczne. W wersji zachłannej — dowolna, większa niż jeden liczba wyrażeń arytmetycznych.

Wynik: liczba.

Przykłady

```
?pisz suma 2 2
4
?pisz (suma 2 3 4 -5)
4
```

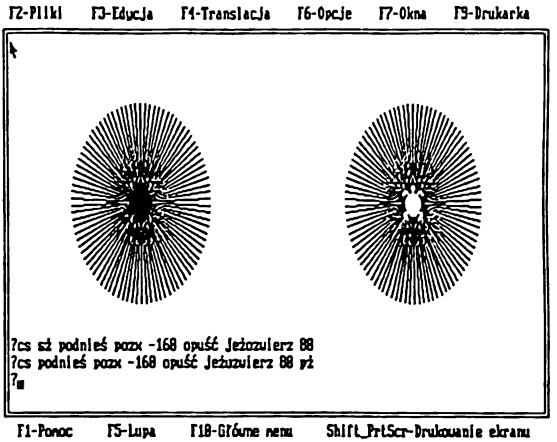
SŻ **HIDETURTLE (HT)**

Składnia: SŻ

Opis działania: procedura "schowaj żółwia", powodująca przejście do takiego trybu rysowania, w którym żółw nie jest widoczny. Tryb ten jest użyteczny, gdy sylwetka żółwia zaciemnia rysunek lub gdy jest celowe zwiększenie szybkości rysowania.

Przykład

```
?oto jeżozwierz :rozmiar
> powtórz 90
>   [
>     naprzód :rozmiar
>     wstecz :rozmiar
>     prawo 4
>   ]
>już
JEŻOZWIERZ zdefiniowano
?czyść
?sż
?podnieś
?poz [ -160 0 ]
?opuść
?jeżozwierz 80
?podnieś
?poz [ 160 0 ]
?opuść
?jeżozwierz 80
?pż
```



Rys. 8.18. Efekty wykonania procedury JEŻOZWIERZ z widoczną sylwetką żółwia (prawy rysunek) oraz bez żółwia (lewy rysunek)

ŚCIER

Patrz ŚCIERANIE

ŚCIERANIE (ŚCIER) PENERASE (PE)

Składnia: ŚCIERANIE

Opis działania: procedura wprowadzająca pisak żółwia w stan "ścieranie". W tym stanie pisak, w trakcie przesuwania żółwia, wymazuje uprzednio wykonane rysunki, czyli pozostawia za sobą ślad w kolorze tła.

Uwaga: wykonanie procedury: OPUŚĆ (ang. PENDOWN), PODNIEŚ (ang. PENUP) lub ODWRACANIE (ang. PENREVERSE) kończy efekt działania procedury ŚCIERANIE. Natomiast wykonanie procedury CZYŚĆ (ang. CLEARSCREEN) i ZMAŻ (ang. CLEAN) nie wpływa na jej działanie.

Przykład

```
?czyść naprzód 40
?ścieranie wstecz 20
{Żółw maże polowę uprzednio narysowanej linii}
```

ŚCIEŻKA

Patrz DYSK

ŚPM

Patrz ŚRODKOWYPRZYCISK

ŚRODKOWYPRZYCISK (ŚPM) **MOUSEMIDDLEBUTTON (MMB)**

Składnia: ŚRODKOWYPRZYCISK → b

Opis działania: funkcja o wartości logicznej PRAWDA, jeśli w momencie wywołania procedury był naciśnięty środkowy przycisk myszki.

Wynik: wartość logiczna.

Uwaga: poprzednie naciśnięcia klawiszy myszki nie są pamiętane (w odróżnieniu od funkcji KLAWISZ?).

Przykład

```
?powtórz 5 [jeśli środkowyprzycisk [ps "tak] inaczej [ps "nie]]
nie
nie
tak          {przyciśnięty środkowy przycisk myszki}
nie          {środkowy przycisk zwolniony}
nie
```

TASUJ **SHUFFLE**

Składnia: TASUJ l → l

Opis działania: funkcja, której wartością jest "potasowana" lista słów będąca jej parametrem; tasowanie polega na losowym przestawianiu kolejnych słów.

Parametr: wyrażenie, którego wartością jest lista zawierająca słowa lub lista pusta.

Wynik: lista powstała z parametru przez pseudolosowe przestawienie elementów.

Przykład

```
?powtórz 4 [pisz tasuj [nie ma róży bez kolców]]
```

kolców nie ma bez róży
ma róży kolców bez nie
kolców róży nie ma bez
nie bez ma róży kolców

TEKSTY (TS) **TEXTSCREEN (TS)**

Składnia: TEKSTY

Opis działania: procedura powodująca przeznaczenie całego ekranu na okienko tekstowe oraz wymazanie jego zawartości. Rysunki z okienka graficznego są gubione.

TG **TAN**

Składnia: TG n → n

Opis działania: funkcja trygonometryczna tangens.

Parametr: wyrażenie, którego wartością jest kąt w stopniach.

Wynik: liczba równa tangensowi parametru.

Przykład

```
?pisz tg 45
1
```

TŁO **BACKGROUND (BG)**

Składnia: TŁO n

Opis działania: procedura ustalająca kolor tła pisaka żółwia. Po wykonaniu procedury TŁO kolory ekranu nie ulegają zmianie. Kolor tła pisaka żółwia będzie nadawany tym punktom, nad którymi zostanie przesunięty żółw z pisakiem w stanie "odwracanie" lub "ścieranie".

Parametr: wyrażenie, którego wartością jest liczba z zakresu (0, 15), będąca numerem koloru. Liczby niecałkowite są zaokrąglane. Liczby większe od 15 są niejawnie przekształcane do zakresu (0, 15) za pomocą operacji modulo 16.

Przykład

```
?tlo 2
?naprzód 10
{Kolor tla nie ulega zmianie}
```

?czyść
{Kolor tla calego okna graficznego zmienia się na zielony}

TŁOTEKSTU (TT) **TEXTBACKGROUND (TG)**

Składnia: TŁOTEKSTU n

Opis działania: procedura ustalająca kolor tla dla znaków pisanych w oknie tekstowo-graficznym. Po wykonaniu procedury TŁOTEKSTU kolory ekranu nie ulegają zmianie.

Parametr: wyrażenie, którego wartością jest liczba z zakresu (0, 15), będąca numerem koloru. Liczby niecalkowite są zaokrąglane. Liczby większe od 15 są niejawnie przekształcane do zakresu (0, 15) za pomocą operacji modulo 16.

Przykład
```
?cs
?tlotekstu 2
{Kolor tla calego okienka tekstowego nie ulega zmianie}
?zt
{Kolor tla calego okienka tekstowego zmienia się na zielony}
```

TREŚĆ **TEXT**

Składnia: TREŚĆ s → l

Opis działania: funkcja, której wartością jest lista zawierająca treść procedury o nazwie podanej jako parametr.

Parametr: wyrażenie, którego wartością jest slowo.

Wynik: lista zawierająca dwie podlisty. Jeżeli parametr jest nazwą zadeklarowanej procedury, to wartością pierwszej z nich są parametry formalne, a drugiej — treść procedury odtworzona na podstawie jej postaci półskompilowanej. Jeżeli parametr nie jest nazwą procedury, to wartością funkcji jest lista złożona z dwóch pustych podlist.

Uwagi: procedura może być zadeklarowana zarówno procedurą OTO (ang. TO), jak i procedurą OKREŚL (ang. DEFINE).
Lista będąca wartością procedury TREŚĆ nie zawiera komentarzy oraz nadmiarowych separatorów użytych do formatowania tekstu źródlowego. Procedury TREŚĆ wraz z procedurą OKREŚL można użyć w celu zmniejszenia wielkości pamięci zajmowanej przez reprezentację wewnętrzną procedur (usunięcie tekstu źródlowego wybranych procedur).

Przykłady

```
?oto pięciokąt :bok
> lewo 18 {wyrównanie położenia}
> powtórz 5
>   [naprzód :bok prawo 72]
> prawo 18
>już
PIĘCIOKĄT zdefiniowano
?pokaż treść "pięciokąt
[ [ :bok ] [lewo 18 powtórz 5 [naprzód :bok prawo 72 ] prawo 18 już ]
?określ "pięciokąt treść "pięciokąt
?pow

?ponp
OTO "pięciokąt :bok

?oto a
> pisz treść "a
> usuń "a
> pisz treść "a
>już
A zdefiniowano
?a
[ ] [ pisz treść "a usuń "a pisz treść "a już ]
[ ] [ ]
```

TRYBDRUKU **SETCOPYMODE (SCM)**

Składnia: TRYBDRUKU n

Opis działania: procedura ustalająca typ drukarki oraz numer trybu graficznego drukarki, używanego podczas wydruku kopii ekranu. Są dostępne następujące tryby pracy:

poniżej 30 — drukarka 8-igłowa,
od 30 do 74 — drukarka 24-igłowa,
powyżej 74 — drukarka laserowa.

Podane parametry są zaokrąglane do najbliższych możliwych; na przykład dla drukarki laserowej są dostępne tryby 75, 100, 150 i 300. Dokładne znaczenie numerów trybu graficznego należy sprawdzić w dokumentacji danej drukarki.

Parametr: liczba nieujemna.

Uwagi: tryb drukarki można zmieniać także przy pomocy poleceń z pliku konfiguracyjnego (por. Dodatek F).

TRYBWZORU **FILLTYPE**

Składnia: TRYBWZORU n

Opis działania: procedura ustalająca sposób działania procedury ZAMALUJ (ang. FILL) oraz sposób rysowania wnętrza figur graficznych (procedury .KOŁO, .ELIPSA, .KWADRAT itp.). Są dostępne następujące sposoby zamalowywania:

0 — zamalowywanie jednolitą "plamą" w kolorze atramentu, granica zamalowywania jest określona przez dowolny kolor nie będący aktualnie zdefiniowanym kolorem tła dla żółwia,

1 — zamalowywanie wzorem ustalonym za pomocą procedury WZÓR z jednoczesnym sumowaniem z poprzednią zawartością pamięci ekranu, granica zamalowywania jest określona przez dowolny kolor nie będący aktualnie zdefiniowanym kolorem tła dla żółwia,

2 — zamalowywanie wzorem ustalonym za pomocą procedury WZÓR (ang. FILLPATTERN) z usuwaniem poprzedniej zawartości pamięci ekranu, granica zamalowywania jest określona przez dowolny kolor nie będący aktualnie zdefiniowanym kolorem tła dla żółwia,

3 — rysowanie tylko obwodu figury, bez wypełniania jej wnętrza wzorem lub plamą.

4 — zamalowywanie jednolitą "plamą" w kolorze atramentu, granica zamalowywania jest określona przez kolor będący aktualnie zdefiniowanym kolorem atramentu dla żółwia,

5 — zamalowywanie wzorem ustalonym za pomocą procedury WZÓR z jednoczesnym sumowaniem z poprzednią zawartością pamięci ekranu, granica zamalowywania jest określona przez kolor będący aktualnie zdefiniowanym kolorem atramentu dla żółwia,

6 — zamalowywanie wzorem ustalonym za pomocą procedury WZÓR (ang. FILLPATTERN) z usuwaniem poprzedniej zawartości pamięci ekranu, granica zamalowywania jest określona przez kolor będący aktualnie zdefiniowanym kolorem atramentu dla żółwia,

Parametr: liczba o wartości 0, 1, 2, 3, 4, 5 lub 6

Uwagi: dla procedury ZAMALUJ sposoby numer 1 i 2 oraz 4 i 5 są parami równoważne, natomiast sposób numer 3 powoduje wyłączenie jej działania.

TS

Patrz TEKSTY

TT

Patrz TŁOTEKSTU

US

Patrz USUŃ

USUŃ (US) **ERASE (ER)**

Składnia: USUŃ s

Opis działania: procedura usuwająca z pamięci procedurę o podanej nazwie.

Parametr: wyrażenie, którego wartością jest słowo będące nazwą procedury.

Uwaga: można usunąć z pamięci aktualnie wykonywaną procedurę, jednak jej postać półskompilowana będzie nadal przechowywana, aż do zakończenia wykonywania tej procedury.

Przykład

```
?oto siedmiokąt :bok
>powtórz 7
> [ naprzód :bok prawo 360 / 7 ]
>już
SIEDMIOKĄT zdefiniowano
?przypisz "8kąt [[:bok] [powtórz 8 [naprzód :bok prawo 45] już]]
?określ "ośmiokąt :8kąt
?usuń siedmiokąt
Błędny parametr USUŃ
?usuń "siedmiokąt
{Usunięcie tekstu źródłowego i półskompilowanego procedury SIEDMIOKĄT}
?usuń "ośmiokąt
{Usunięcie tekstu półskompilowanego procedury OŚMIOKĄT}
```

USUŃPLIK **ERASEFILE (ERF)**

Składnia: USUŃPLIK s

Opis działania: procedura usuwająca z dysku plik o podanej nazwie.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku.

Przykład

```
?usuńplik "b:ala.doc
{Usunięcie z dysku B: pliku ala.doc}
?usuńplik "ola.doc
{Usunięcie z dysku i katalogu bieżącego pliku ola.doc}
```

USW **ERALL**

Składnia: USW

Opis działania: procedura "usuń wszystko", usuwająca z pamięci wszystkie zadeklarowane procedury i zmienne.

Przykład

```
?oto wyświetl :x
> pisz :x
>już
WYŚWIETL zdefiniowano
?wyświetl "Alinka
Alinka
?przypisz "złotówka "moneta
?pisz :złotówka
moneta
?usw
{usunięcie z pamięci tekstów wszystkich procedur i zmiennych}
?wyświetl "Alinka
nie wiem jak zrobić WYŚWIETL
?pisz :złotówka
ZŁOTÓWKA nie ma zdefiniowanej wartości.
?oto autoexec
> ps "" *** To jest LOGO Krzysia ***'
> usw
>już
AUTOEXEC zdefiniowano
?zapisz "autoexec
{por. rozdział 4}
```

USWP **ERPS**

Składnia: USWP

Opis działania: procedura "usuń wszystkie procedury", usuwająca z pamięci operacyjnej wszystkie zadeklarowane procedury.

Przykład

```
?przypisz "obiad [[zupa] [drugie danie] [deser]]
?oto obiad
> pisz "zupa
> pisz [drugie danie]
> pisz "deser
>już
OBIAD zdefiniowano
?pisz :obiad
[ zupa ] [ drugie danie ] [ deser ]
?obiad
zupa
drugie danie
deser
?uswp
?pisz :obiad
[ zupa ] [ drugie danie ] [ deser ]
?obiad
Nie wiem jak zrobić OBIAD
```

USWZ **ERNS**

Składnia: USWZ

Opis działania: procedura "usuń wszystkie zmienne", usuwająca z pamięci operacyjnej wszystkie zmienne globalne.

Przykłady

```
?przypisz "spis [ Andrzej Jarek Krzyś ]
?pisz :spis
Andrzej Jarek Krzyś
?uswz
?pisz :SPIS
SPIS nie ma wartości
?oto procedura :a
> lokalna "b
> przypisz "b 2
> uswz
```

```
> pisz :a    {A jest zmienną lokalną — parametrem}
> pisz :b    {B jest zmienną lokalną}
> pisz :c    {C jest zmienną globalną}
>już
PROCEDURA zdefiniowano
?przypisz "c 3
?procedura1
1
2
C nie ma zdefiniowanej wartości na poziomie 1 w procedurze PROCEDURA
```

USZ **ERN**

Składnia: USZ s

Opis działania: procedura "usuń zmienną", usuwająca z pamięci operacyjnej zmienną, której nazwa jest parametrem.

Parametr: wyrażenie, którego wartością jest słowo.

Przykłady

```
?przypisz "pomarańcza "duża
?przypisz "miodek "słodki
?pisz :pomarańcza
duża
?pisz :miodek
słodki
?usz "miodek
?pisz :pomarańcza
duża
?pisz :miodek
MIODEK nie ma zdefiniowanej wartości.
```

WAR

Patrz WARTOŚĆ

WARTOŚĆ **THING**

Składnia: WARTOŚĆ s → sl

Opis działania: funkcja, której wartością jest aktualna wartość zmiennej o nazwie podanej jako parametr.

Parametr: wyrażenie, którego wartością jest słowo będące nazwą zmiennej.

Wynik: słowo lub lista będąca wartością zmiennej.

Przykłady

```
?przypisz "wzrost 178
?pisz :wzrost
178
?pisz wartość "wzrost
178
?przypisz "Magda "żona
?przypisz "żona "kobieta
?przypisz "kobieta [puch marny]
?pisz :Magda
żona
?pisz wartość "Magda
żona
?pisz wartość :Magda
kobieta
?pisz wartość wartość "Magda
kobieta
?pisz wartość wartość wartość "Magda
puch marny
```

WL **UPPERCASE (UC)**

Składnia: WL s → s

Opis działania: funkcja "wielkie litery", której wartością jest słowo powstałe z parametru przez zamianę małych liter na wielkie.

Parametr: wyrażenie, którego wartością jest słowo.

Wynik: słowo nie zawierające małych liter.

Uwaga: w wyniku działania funkcji nie ulegają zmianie znaki nie będące literami.

Przykłady

```
?pisz wl "Iwona
IWONA
?pisz wl "znakiI"#$%&
ZNAKII#$%&
?oto imię :i
> wynik słowo wl pierw :i bezpierw :i
>już
```

```
IMIĘ zdefiniowano
?pisz imię "alina
Alina
```

WŁĄCZDRUK — **PRINTON (PON)**

Składnia: WŁĄCZDRUK

Opis działania: procedura powodująca kopiowanie na drukarce tekstów wyprowadzanych na ekran. Procedura kończy działanie po wykonaniu procedury WYŁĄCZDRUK (ang. PRINTOFF).

WŁĄCZŚLAD — **TRACE**

Składnia: WŁĄCZŚLAD

Opis działania: procedura powodująca włączenie śledzenia programu. Śledzenie polega na wyświetlaniu dodatkowych informacji w czasie działania programu oraz na wyświetlaniu w specjalnym okienku tekstu aktualnie wykonywanej procedury. Sygnalizowane są momenty rozpoczęcia i zakończenia wykonywania procedur oraz przypisania wartości zmiennym. Sygnalizacja rozpoczęcia wykonywania procedury polega na wyświetleniu w nawiasach kwadratowych poziomu wywołania oraz nagłówka procedury z parametrami formalnymi zastąpionymi przez aktualne. Sygnalizacja zakończenia wykonywania procedury polega na wyświetleniu w nawiasach kwadratowych poziomu wywołania, słowa JUŻ (ang. END), nazwy procedury oraz w przypadku funkcji jej wartości poprzedzonej słowem WYNIK (ang. OUTPUT). Sygnalizacja przypisania wartości zmiennej polega na wyświetleniu w nawiasach kwadratowych poziomu aktualnie wykonywanej procedury, słowa PRZYPISZ (ang. MAKE), nazwy zmiennej oraz przypisywanej jej wartości. Śledzenie jest pomocne w wyszukiwaniu i usuwaniu błędów z programów. Do wyłączenia śledzenia służy procedura WYŁĄCZŚLAD (ang. UNTRACE).

Uwagi: jeżeli są otwarte okienka śledzenia (por. rozdział 7), to informacje o śledzeniu pracy programu są wyświetlane w tych okienkach (w jednym z nich jest wyświetlany tekst aktualnie wykonywanej procedury). Jeżeli okienka śledzenia są zamknięte, to informacje o śledzeniu programu pojawiają się w okienku tekstowym — nie jest wtedy wyświetlany tekst aktualnie wykonywanej procedury.

Przykłady

```
?oto procedura
> przypisz "zmienna 1
> przypisz "zmienna funkcja :zmienna
> przypisz "zmienna 3
>już
PROCEDURA zdefiniowano
?oto funkcja :x
> przypisz "zmienna 2
> wynik 10 * :x
>już
FUNKCJA zdefiniowano
?włączślad procedura
[1] OTO procedura
[1] PRZYPISZ "ZMIENNA 1
[2] OTO funkcja 1
[2] PRZYPISZ "ZMIENNA 2
[2] JUŻ funkcja WYNIK 10
[1] PRZYPISZ "ZMIENNA 10
[1] PRZYPISZ "ZMIENNA 3
[1] JUŻ procedura
?oto fib :n
> jeśli :n < 3
>   [wynik 1]
> inaczej
>   [wynik (fib :n - 1) + (fib :n - 2)]
>już
FIB zdefiniowano
?wyłączślad pisz fib 4
3
?włączślad pisz fib 4
[1] OTO fib 4
[2] OTO fib 3
[3] OTO fib 2
[3] JUŻ fib WYNIK 1
[3] OTO fib 1
[3] JUŻ fib WYNIK 1
[2] JUŻ fib WYNIK 2
[2] OTO fib 2
[2] JUŻ fib WYNIK 1
[1] JUŻ fib WYNIK 3
?oto cenzura :zdanie :slowo
> jeśli puste? :zdanie [pisz [] stop]
> jeśli równe? :slowo ml pierw :zdanie
>   [wpisz "...\ ]
```

```
> inaczej
>   [(wpisz pierw :zdanie "\ )]
> cenzura bezpierw :zdanie :slowo
>już
CENZURA zdefiniowano
?wyłączślad cenzura [To jest kot Ali.] "kot
To jest ... Ali.
?włączślad cenzura [To jest kot Ali.] "kot
[1] OTO cenzura [ To jest kot Ali. ] "kot
To [2] OTO cenzura [ jest kot Ali. ] "kot
{Dalej stale poziom [3] ze względu na optymalizację rekurencji ogonowej}
jest [3] OTO cenzura [ kot Ali. ] "kot
Ali. [3] OTO cenzura [ Ali. ] "kot
[3] JUŻ cenzura
[2] JUŻ cenzura
[1] JUŻ cenzura
```

WOLNE **.FREESPACE**

Składnia: WOLNE → n

Opis działania: funkcja, której wartością jest liczba dostępnych bajtów pamięci w obszarze pamięci obiektów (obiektów tworzonych przez użytkownika).

Wynik: liczba.

Uwagi: rzeczywista wielkość pamięci dla całego systemu LOGO może być wyświetlona przy pomocy systemu okienkowego — obszar obiektów nie obejmuje definicji procedur, zmiennych oraz obszarów roboczych edytora i systemu okienkowego. Do obszaru obiektów nie jest także wliczany obszar zadeklarowany za pomocą procedury .ZAJMIJ (ang. .RESERVE).

Przykład

```
?pisz wolne
59027
```

WPISZ **TYPE**

Składnia: WPISZ sl

Opis działania: procedura powodująca wyświetlenie na ekranie obiektu będącego jej parametrem. Znak cudzysłowu " poprzedzający słowo oraz zewnętrzne nawiasy listy nie są wyświetlane. Wydruk nie jest uzupełniany znakiem nowego wiersza.

Parametr: dowolne wyrażenie.

Uwagi: jeżeli bezpośrednio po wykonaniu procedury WPISZ nastąpi powrót do interpretatora poleceń, to znak zgłoszenia edytora wierszowego zostanie wyświetlony bezpośrednio za wyprowadzonym tekstem.
Słowo puste lub lista pusta nie są drukowane.

Przykłady

```
?wpisz [1 + 2 =] pisz 1 + 2
1 + 2 = 3
?oto wyświetl :warunek
> wpisz :warunek
> wpisz [\ -\ warunek\ ]
> jeśli :warunek
>   [pisz [ jest spełniony]]
> inaczej
>   [pisz [ nie jest spełniony]]
>już
WYŚWIETL zdefiniowano
?wyświetl 1 = 5
FAŁSZ — warunek nie jest spełniony
?oto tabela_kwadratów :n
> lokalna "i
>
> przypisz "i 1
> powtórz :n
> [
>   wpisz :i
>   wpisz [\ do kwadratu równa się ]
>   pisz :i * :i
>   przypisz "i :i + 1
> ]
>już
TABELA_KWADRATÓW zdefiniowano
?tabela_kwadratów 5
1 do kwadratu równa się 1
2 do kwadratu równa się 4
3 do kwadratu równa się 9
4 do kwadratu równa się 16
5 do kwadratu równa się 25
```

WRÓĆ **HOME**

Składnia: WRÓĆ

Opis działania: procedura ustawiająca żółwia w położeniu początkowym, o współrzędnych x = 0, y = 0, skierowanego pionowo w górę.

Przykład

```
?wróć
?pisz poz?
0 0
?pisz kąt?
0
```

WS

Patrz WSTECZ

WSTECZ **BACK (BK)**

Składnia: WSTECZ n

Opis działania: procedura przesuwająca żółwia o zadaną liczbę kroków do tyłu.

Parametr: wyrażenie arytmetyczne określające liczbę kroków.

Uwagi: przy ujemnej wartości parametru następuje przesunięcie żółwia do przodu.
Standardowo, krokiem żółwia jest odległość między dwoma kolejnymi punktami ekranu w pionie i poziomie. Krok można zmieniać za pomocą procedury PROPORCJA (ang. SETSCRUNCH).

Przykład

```
?wstecz 20 * 2
{Przesunięcie żółwia o 40 kroków do tyłu}
?wstecz -50
{Przesunięcie żółwia o 50 kroków do przodu}
```

WY

Patrz WYNIK

WYŁĄCZDRUK **PRINTOFF (POF)**

Składnia: WYŁĄCZDRUK

Opis działania: procedura powodująca, że teksty wyprowadzane na ekran nie są kopiowane na drukarce. Jej działanie jest odwrotne do działania procedury WŁĄCZDRUK (ang. PRINTON).

WYŁĄCZŚLAD **UNTRACE**

Składnia: WYŁĄCZŚLAD

Opis działania: procedura o działaniu odwrotnym do działania procedury WŁĄCZŚLAD (ang. TRACE).

WYNIK (WY) **OUTPUT (OP)**

Składnia: WYNIK sl

Opis działania: procedura powodująca zakończenie działania funkcji i przekazanie wartości jej parametru do procedury wywołującej. Procedura może być użyta tylko w treści funkcji.

Parametr: dowolne wyrażenie.

Przykłady

```
?oto e
> wynik 2.71828
>już
?pisz e
2.71828
?oto silnia :n
> jeśli :n < 3
>    [wynik :n]
> wynik :n * silnia :n - 1
>już
SILNIA zdefiniowano
?pisz silnia 1
1
?pisz silnia 3
6
?pisz silnia 10
3628800
```

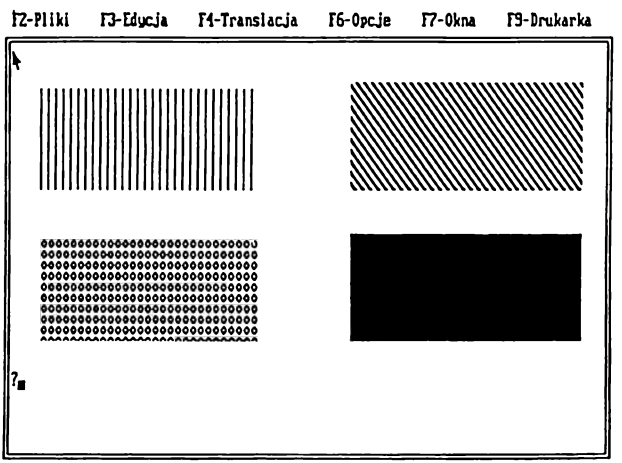
WZÓR — FILLPATTERN

Składnia: WZÓR l

Opis działania: procedura pozwalająca określić wzór wypełniania używany w procedurze ZAMALUJ (ang. FILL). Deklarowany jest wzorzec o rozmiarze 8 * 8 punktów. Podczas zamalowywania obszaru, każdemu punktowi jest nadawany kolor odpowiadającego mu punktu wzorca.

Parametr: wyrażenie, którego wartością jest ośmioelementowa lista postaci [n n n n n n n n]. Jej elementy są liczbami z zakresu (0, 255). Liczby niecałkowite są zaokrąglane. Każda liczba określa stan ośmiu punktów wzorca należących do jednej linii poziomej. Pierwsza liczba określa stan najwyższej linii. Kolejne pozycje binarne liczby określają stan kolejnych punktów: jedynka oznacza kolor atramentu, zero kolor tła.

Uwaga: po wywołaniu systemu programowania AC–LOGO, przed pierwszym wywołaniem procedury WZÓR, przyjmowany jest wzór standardowy — jednolita plama w kolorze atramentu.



Rys. 8.19. Przykłady okna graficznego wypełnionego różnymi wzorami

Przykłady

```
?wzór [#80 #80 #80 #80 #80 #80 #80 #80]
{80 szesnastkowo odpowiada 10000000 binarnie}
?czyść
?zamaluj
{Wypełnienie całego okna graficznego liniami pionowymi}
?wzór [0 #18 #24 #42 #42 #24 #18 0]
?czyść
?zamaluj
{Wypełnienie całego okna graficznego kólkami}
?wzór {#F0 #78 #3C #1E #0F #87 #C3 #E1]
?czyść
?zamaluj
{Wypełnienie całego okna graficznego grubymi liniami ukośnymi}
?wzór [255 255 255 255 255 255 255 255]
{Powrót do wzoru standardowego}
```

WZÓR16 **FILLPATT16**

Składnia: WZÓR16 I

Opis działania: procedura pozwalająca określić wzór wypełniania używany w procedurze ZAMALUJ (ang. FILL). Deklarowany jest wzorzec o rozmiarze 16 * 16 punktów. Podczas zamalowywania obszaru, każdemu punktowi jest nadawany kolor odpowiadającego mu punktu wzorca.

Parametr: wyrażenie, którego wartością jest szesnastoelementowa lista postaci [n n . . . n]. Jej elementy są liczbami z zakresu (0, 65535). Liczby niecałkowite są zaokrąglane. Każda liczba określa stan szesnastu punktów wzorca należących do jednej linii poziomej.

Uwagi: procedura ma takie samo działanie, jak procedura WZÓR (ang. FILLPATTERN).
Wzorzec wypełniania definiowany za pomocą procedury WZÓR jest podzbiorem wzorca definiowanego za pomocą WZÓR16 (definiowane pole 8 * 8 punktów jest rozszerzane do pola 16 * 16 punktów przez dwukrotne skopiowanie w kierunku poziomym i pionowym).

WZÓRKRESKI **LINEPATTERN**

Składnia: WZÓRKRESKI n

Opis działania: procedura pozwalająca określić wzorzec bitowy kreski rysowanej przez pisak żółwia. Deklarowany jest wzorzec o rozmiarze 16 bitów (punktów). Podczas rysowania kreski, każdemu punktowi wzoru będącemu "1" odpowiada narysowanie punktu w kolorze atramentu.

Parametr: wyrażenie liczbowe o wartości z zakresu (0, 65535). Wartości 0 odpowiada brak rysowania kreski przez pisak żółwia, natomiast wartości 65535 — rysowanie kreski ciągłej.

Uwagi: po wczytaniu systemu AC–LOGO jest przyjmowany wzorzec kreski 65535, odpowiadający kresce ciągłej.

Przykłady

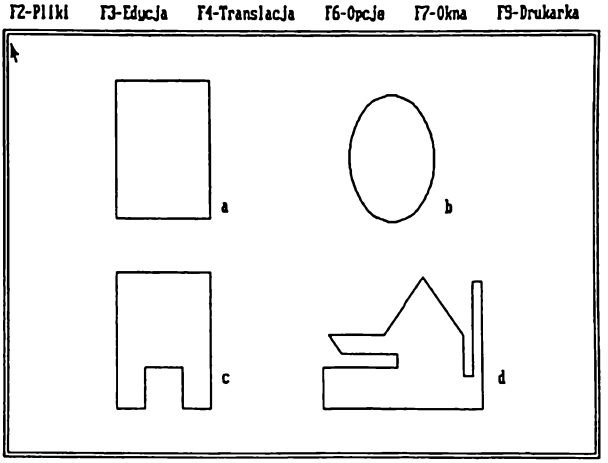
```
?wzórkreski #ffff naprzód 25
?wzórkreski #8888 naprzód 25
?wzórkreski #cccc naprzód 25
{na ekranie pojawia się kreska ciągła, następnie kreska "kropkowa" oraz kreska przerywana}
```

ZAMALUJ **FILL**

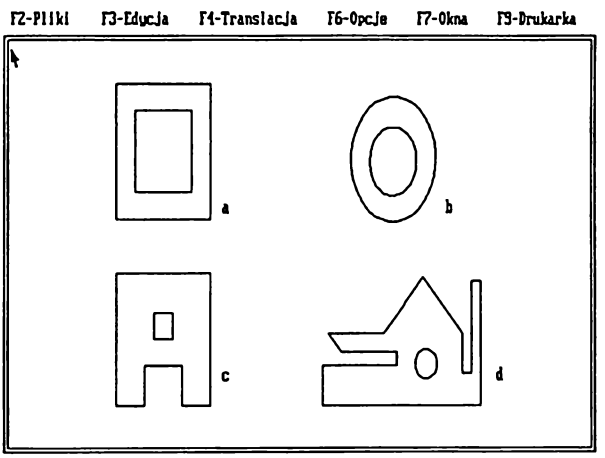
Składnia: ZAMALUJ

Opis działania: procedura wypełniająca dowolny zamknięty obszar wzorem określonym za pomocą ostatniego wywołania procedury WZÓR (ang. FILLPATTERN) lub WZÓR16 (ang. FILLPATT16), poczynając od bieżącej pozycji żółwia. Przez obszar zamknięty należy rozumieć spójny obszar złożony z punktów w kolorze tła, należących do okna graficznego.

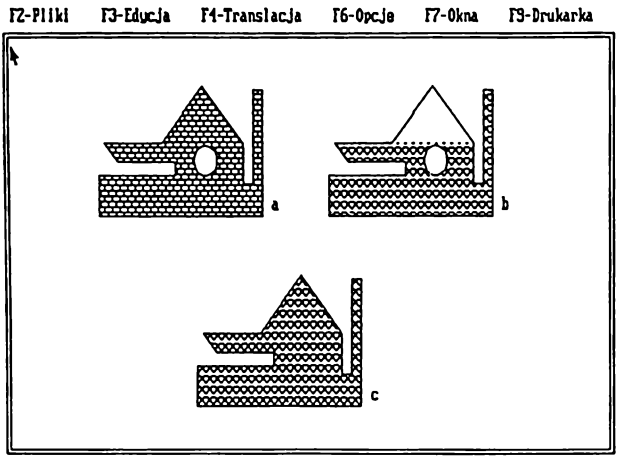
Uwagi: obszary o brzegu będącym figurą spójną (por. rysunek 8.20) mogą być wypełniane dowolnym wzorem. Obszary o brzegu nie będącym figurą spójną (por. rysunek 8.21) mogą być wypełniane tylko takim wzorem, aby po wypełnieniu obszaru punkty w kolorze atramentu stanowiły figurę spójną (por. rysunek 8.22). W przypadku użycia niewłaściwego wzoru następuje zapętlenie się procedury ZAMALUJ i zawieszenie komputera na okres kilku sekund. Czas wypełniania obszarów zależy od rodzaju wzoru — im więcej punktów w kolorze atramentu zawiera wzór tym wypełnianie obszaru jest szybsze. Wypełniania nie można rozpocząć w punktach graniczących z krawędzią okna graficznego.



Rys. 8.20. Przykłady obszarów o brzegach będących figurą spójną



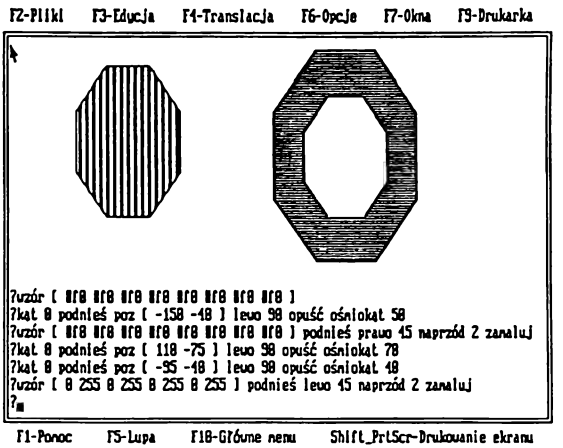
Rys. 8.21. Przykłady obszarów o brzegach nie będących figurą spójną



Rys. 8.22. Efekty wypełniania obszarów wzorem:
a) wypełnianie obszaru z rysunku 8.21d właściwym wzorem;
b) wypełnianie obszaru z rysunku 8.21d niewłaściwym wzorem;
c) obszar z rysunku 8.20d można wypełnić dowolnym wzorem

Przykłady

```
?oto ośmiokąt :bok
> powtórz 8 [ naprzód :bok prawo 45 ]
>już
OŚMIOKĄT zdefiniowano
?czyść sż
?wzór [#F0 #F0 #F0 #F0 #F0 #F0 #F0 #F0]
?lewo 90 ośmiokąt 30
?podnieś prawo 45 naprzód 2 zamaluj
{Wypełnienie ośmiokąta grubymi pionowymi liniami}
?czyść wzór [0 255 0 255 0 255 0 255]
?lewo 90 opuść ośmiokąt 20
?podnieś poz [14 -31] opuść ośmiokąt 45
?podnieś prawo 45 naprzód 2
?zamaluj
{Wypełnienie obszaru między ośmiokątami poziomymi liniami}
```



Rys. 8.23. Przykłady wypełniania obszarów za pomocą procedury pierwotnej ZAMALUJ

ZAPISZ — SAVEALL

Składnia: ZAPISZ s

Opis działania: procedura powodująca zapisanie treści wszystkich procedur zadeklarowanych za pomocą procedury OTO (ang. TO) w pliku tekstowym o nazwie będącej parametrem.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeżeli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie LOG (rozszerzenie to można zmienić na inne za pomocą poleceń pliku konfiguracyjnego — por. Dodatek F).

Przykład

```
?teksty usw
?oto trójkąt :bok
> powtórz 3 [naprzód :bok prawo 120]
>już
TRÓJKĄT zdefiniowano
?oto n_kąt :n :bok
>{wielokąt foremny o :n kątach}
> powtórz :n [naprzód :bok prawo 360 / :n]
>już
```

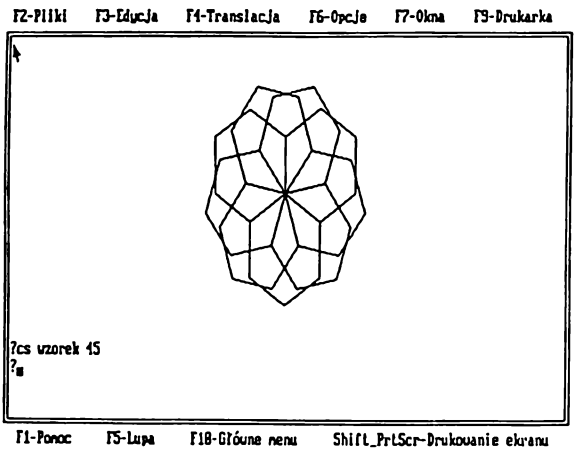
N_KĄT zdefiniowano
?zapisz "b:wielobok
{Zapisanie na dysku B:, w pliku o nazwie wielobok.log tekstów źródłowych procedur trójkąt i n_kąt}

ZAPISZO **SAVESCR**

Składnia: ZAPISZO s

Opis działania: procedura "zapisz obraz", powodująca zapisanie aktualnej zawartości pamięci okna graficzno-tekstowego, w pliku o nazwie podanej jako parametr.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeżeli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie COD (rozszerzenie to można zmienić za pomocą poleceń pliku konfiguracyjnego — por. Dodatek F).



Rys. 8.24. Efekt wykonania procedury WZOREK

Przykład

```
?oto wzorek :bok
> grafika
> powtórz 9
>  [
```

```
>    powtórz 6 [naprzód :bok prawo 60] prawo 40
>  ]
> zapiszo "wzorek
>już
WZOREK zdefiniowano
?wzorek 45
```

ZAPISZOKNO **SAVEWND**

Składnia: ZAPISZOKNO s

Opis działania: procedura "zapisz okno", powodująca zapisanie aktualnej zawartości pamięci okienka graficznego, w pliku o nazwie podanej jako parametr.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeżeli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie COD (rozszerzenie to można zmienić za pomocą poleceń pliku konfiguracyjnego — por. Dodatek F).

ZAPISZP **SAVE**

Składnia: ZAPISZP s sl

Opis działania: procedura "zapisz procedury", powodująca zapisanie w pliku o nazwie będącej pierwszym parametrem, treści procedur, których nazwy podano jako drugi parametr.

Parametry: pierwszy parametr jest wyrażeniem, którego wartością jest słowo będące jednoznaczną nazwą pliku (jeśli podana nazwa nie zawiera rozszerzenia to dodawane jest rozszerzenie LOG). Drugi parametr jest wyrażeniem, którego wartością jest słowo lub lista postaci [s s ...], gdzie zarówno słowo, jak i elementy listy są nazwami procedur.

Uwaga: treść procedur zadeklarowanych procedurą OKREŚL (ang. DEFINE) nie może być zapisywana za pomocą procedury ZAPISZP.

Przykłady

```
?zapiszp "b:ala "ala
{Zapisanie na dysku B:, w pliku o nazwie ala.log procedury ala}
?zapiszp "malunki [maluj namaluj]
{Zapisanie na dysku bieżącym, w pliku o nazwie malunki.log procedur maluj
i namaluj}
```

ZAPISZRED **SAVED**

Składnia: ZAPISZRED s

Opis działania: procedura "zapisz edytor", powodująca zapisanie w pliku o nazwie będącej pierwszym parametrem aktualnej zawartości obszaru roboczego edytora.

Parametr: wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeżeli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie LOG (rozszerzenie to można zmienić za pomocą poleceń pliku konfiguracyjnego — por. Dodatek F).

Przykłady

```
?redw
{porzucenie redagowania za pomocą polecenia ^K–Q}
?zapiszred "edytor
{zapisanie treści wszystkich procedur w pliku EDYTOR.LOG}
```

ZAPISZW **SAVEOBJ**

Składnia: ZAPISZW s sl

Opis działania: procedura "zapisz wartość", powodująca zapisanie w pliku o nazwie będącej pierwszym parametrem, obiektu podanego jako drugi parametr, w postaci tekstowej.

Parametry: pierwszy parametr jest wyrażeniem, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeśli podana nazwa nie zawiera rozszerzenia to dodawane jest rozszerzenie TXT. Drugi parametr jest dowolnym wyrażeniem.

Uwaga: zewnętrzne nawiasy listy nie są zapisywane do pliku.

Przykład

```
?przypisz "przeliczniki [1.67 0.71 11.82]
?zapiszw "b:dane :przeliczniki
{Zapisanie tekstu: "1.67 0.71 11.82" na dysku B:, w pliku o nazwie dane.txt}
?zapiszw "spis.lst zdanie [Ala Ola] [Jan Adam]
{Zapisanie tekstu: "Ala Ola Jan Adam" w pliku o nazwie spis.lst }
?pisz ładujw "b:dane.txt
 1.67 0.71 11.82
?pokaż ładujw "spis.lst
[ Ala Ola Jan Adam ]
```

ZD

Patrz ZDANIE

ZDANIE (ZD) **SENTENCE (SE)**

Składnia: ZDANIE sl sl → l
lub: (ZDANIE sl sl ...) → l

Opis działania: funkcja, której wartością jest lista utworzona z obiektów podanych jako parametry. Jeśli obiekt jest listą, to jej zewnętrzne nawiasy są usuwane.

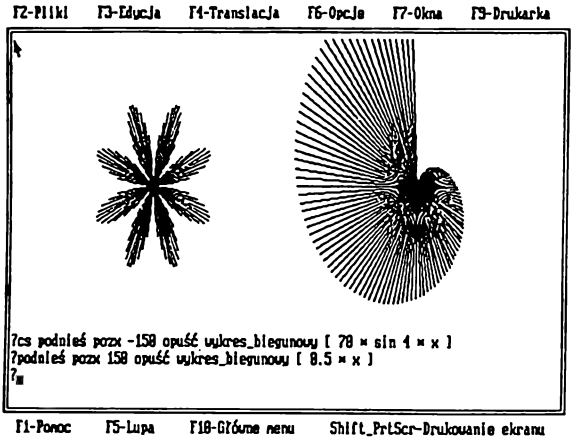
Parametry: dwa dowolne wyrażenia. W wersji zachłannej — od 2 do 127 dowolnych wyrażeń.

Wynik: lista.

Przykłady

```
?pokaż zdanie 1 2
[ 1 2 ]
?pokaż zdanie [1 2] "Ala
[ 1 2 Ala ]
?pokaż zdanie [] []
[ ]
?pokaż zdanie [PRAWDA] 2 > 4
[ PRAWDA FAŁSZ ]
?pokaż zdanie [[Ala][Ola]] [Pola]
[ [ Ala ] [ Ola ] Pola ]
?pokaż (zdanie [Ala] [Ola] [Pola])
[ Ala Ola Pola ]
?pokaż (zdanie "Ala "Ola "Pola)
[ Ala Ola Pola ]
?oto wykres_biegunowy :funkcja
>{parametr jest listą zawierającą wywołanie funkcji —
> przekazanie parametru jest przekazaniem "przez procedurę"}
> sż
> okno
> opuść
> przypisz "x 0
> powtórz 120
> [
>   zrób zdanie [naprzód] :funkcja
>   zrób zdanie [wstecz] :funkcja
>   przypisz "x :x + 3
>   prawo 3
```

```
> ]
>już
WYKRES_BIEGUNOWY zdefiniowano.
?oto x
> wynik :x
>już
X zdefiniowano.
?czyść wykres_biegunowy [ 70 * sin 4 * x ]
?czyść wykres_biegunowy [ 0.5 * x ]
```



Rys. 8.25. Przykłady wykonania procedury WYKRES_BIEGUNOWY

ZMAŻ **CLEAN**

Składnia: ZMAŻ

Opis działania: procedura powodująca wypełnienie obszaru ekranu powyżej okna tekstowego kolorem tła bez zmiany pozycji żółwia. Procedura powoduje usunięcie z ekranu wszystkich znajdujących się na nim rysunków.

Przykład

```
?czyść naprzód 20
?zmaż
?pisz poz?
0 20
```

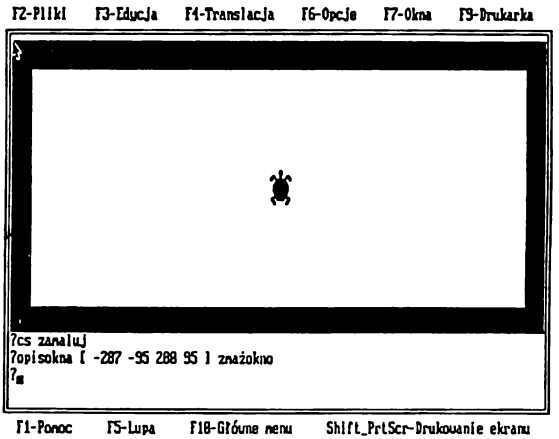
ZMAŻOKNO (ZO) — CLEANWINDOW (CLW)

Składnia: ZMAŻOKNO

Opis działania: procedura powodująca wypełnienie okna graficznego kolorem tła bez zmiany pozycji żółwia, czyli usunięcie z okna graficznego wszystkich znajdujących się na nim rysunków. Przy standardowo zadeklarowanym oknie graficznym procedura ZMAŻOKNO jest równoważna procedurze ZMAŻ (ang. CLEAN).

Przykład

```
?czyść sż
?zamaluj
{Całe okno graficzne zostaje zamalowane kolorem atramentu}
?opisokna [-#6F -#2F #6F #4F] zmażokno
{Na ekranie powstaje ramka okalająca okno graficzne}
```



Rys. 8.26. Przykład zastosowania procedury pierwotnej ZMAŻOKNO

ZMAŻTEKST (ZT) — CLEARTEXT (CT)

Składnia: ZMAŻTEKST

Opis działania: procedura powodująca wymazanie tekstów z okna tekstowego, bez zmiany zawartości okna graficznego ekranu.

ZMIENNA? **NAMEP**

Składnia: ZMIENNA? s → b

Opis działania: funkcja o wartości logicznej PRAWDA, jeśli jej parametr jest nazwą zmiennej, lub FAŁSZ w przeciwnym razie.

Parametr: wyrażenie, którego wartością jest słowo.

Wynik: wartość logiczna.

Przykłady

```
?przypisz "nazwa "wartość
?pisz zmienna? "nazwa
PRAWDA
?pisz zmienna? "wartość
FAŁSZ
?pisz zmienna? :nazwa
FAŁSZ
?przypisz "wartość 1 pisz zmienna? "wartość
PRAWDA
?pisz zmienna? :nazwa
PRAWDA
?usz "nazwa pisz zmienna "nazwa
FAŁSZ
```

ZNAK **CHAR**

Składnia: ZNAK n → s

Opis działania: funkcja, której wartością jest słowo składające się z jednego znaku o podanym kodzie ASCII.

Parametr: wyrażenie arytmetyczne z zakresu (0, 255). Liczby niecałkowite są zaokrąglane. Liczby większe od 255 są niejawnie przekształcane do zakresu (0, 255) za pomocą operacji modulo 256.

Wynik: jednoznakowe słowo.

Przykłady

```
?pisz znak 65
A
?pisz znak 97
a
?pisz znak #31
1
```

ZO

Patrz ZMAŻOKNO

ZRÓB RUN

Składnia: ZRÓB l
lub w wyrażeniach:
ZRÓB l → sl

Opis działania: procedura, która użyta w wyrażeniu jest funkcją, powodująca wykonanie procedur z listy podanej jako parametr.

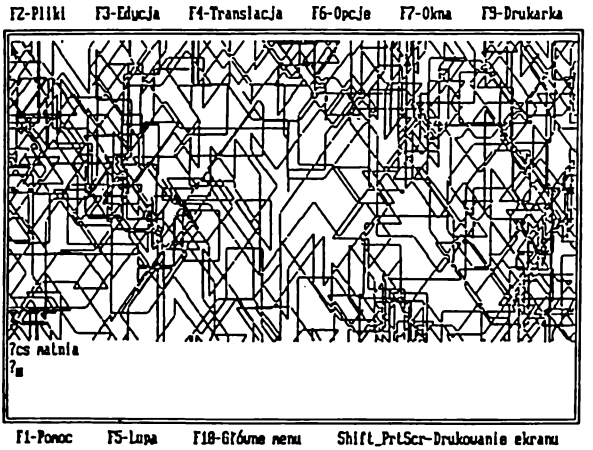
Parametr: wyrażenie, którego wartością jest lista wywołań procedur. W postaci funkcyjnej procedury ZRÓB lista ta zawiera wyrażenie.

Wynik (tylko dla postaci funkcyjnej): slowo lub lista.

Przykłady

```
?zrób [pisz "Alinka]
Alinka
?pisz zrób [1 + 2]
3
?przypisz "kwadrat [powtórz 4 [naprzód 60 prawo 90]]
?zrób :kwadrat
{Narysowanie kwadratu o boku 60}
?oto kalkulator
> lokalna "cl
> przypisz "cl czytajlistę
> jeśli puste? :cl [stop]
> kursor zdanie 0 (ost kursor?) — 1
> (wpisz :cl "" = ")
> zrób zdanie [pisz] :cl
> kalkulator
>już
KALKULATOR zdefiniowano
?kalkulator
2 * 2 = 4
2 + 3 * sin 30 = 3.5
{wciśnięcie klawiszy CTRL i BREAK przerywa działanie procedury}
?oto matnia
> czyść sklej sż odwracanie
> dopóki [nie klawisz?] [naprzód 40 prawo 45 * entier losowa 8 ]
>już
```

```
MATNIA zdefiniowano
?oto dopóki :warunek :lista
>{oba parametry przekazywane "przez procedurę"}
>{:warunek — nazwa funkcji o wartości logicznej}
>{:lista — lista wywołań procedur}
> jeśli zrób zdanie [nie] :warunek [stop]
> zrób :lista dopóki :warunek :lista
>już
DOPÓKI zdefiniowano
?matnia
```



Rys. 8.27. Przykład wykonania procedury MATNIA

ZT

Patrz ZMAŻTEKST

ŻÓŁW? **TURTLESTATE (TST)**

Składnia:	ŻÓŁW? → l
Opis działania:	funkcja, której wartością jest lista określająca stan żółwia.
Wynik:	sześcioelementowa lista postaci [b b n n b b] o następującym znaczeniu elementów:

pierwszy — wartość logiczna: PRAWDA, gdy pisak żółwia jest opuszczony, a FAŁSZ, gdy podniesiony;
drugi — wartość logiczna: PRAWDA, gdy żółw jest widoczny, a FAŁSZ, gdy jest niewidoczny.
trzeci — kolor atramentu — liczba z zakresu (0, 15);
czwarty — kolor tła — liczba z zakresu (0, 15);
piąty — wartość logiczna FAŁSZ;
szósty — wartość logiczna FAŁSZ.

Uwagi: dwa ostatnie elementy listy zostały pozostawione ze względu na zgodność z Junior LOGO, gdzie oznaczały wybór pierwszej (jaskrawej) palety kolorów oraz atrybut migotania. W AC-LOGO nieobecność palet kolorów została zrównoważona większą liczbą dostępnych kolorów. Ze względu na brak realizacji sprzętowej migotania w kartach graficznych EGA/VGA i Hercules HGC mikrokomputera IBM-PC atrybut migotania nie jest dostępny.

Przykłady

```
?czyść
?pokaż żółw?
[ PRAWDA PRAWDA 7 1 FAŁSZ FAŁSZ ]
?okno naprzód 200
?pokaż żółw?
[ PRAWDA PRAWDA 7 1 FAŁSZ FAŁSZ ]
{Żółw jest poza oknem, ale pomimo tego jest w stanie "widoczny"}
?czyść sż
?pokaż żółw?
[ PRAWDA FAŁSZ 7 1 FAŁSZ FAŁSZ ]
?podnieś
?pokaż żółw?
[ FAŁSZ FAŁSZ 7 1 FAŁSZ FAŁSZ ]
{pisak żółwia jest podniesiony}
```

Składnia: .ELIPSA l

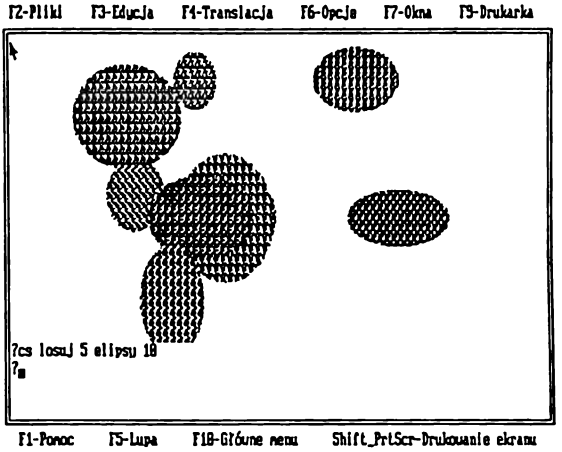
Opis działania: narysowanie elipsy.

Parametr: lista postaci [x y a b], gdzie x i y są liczbami z zakresu (–32000, 32000) oznaczającymi współrzędne środka elipsy, a a i b są liczbami z zakresu (0, 32000) oznaczającymi promienie elipsy.

Uwagi: sposób wypełniania wnętrza figury zależy od sposobu i wzoru wypełniania, zdefiniowanych za pomocą TRYBWZORU (ang. FILLTYPE) i WZÓR lub WZÓR16 (ang. FILLPATTERN, FILLPATT16).

Przykład

```
?oto "elipsy :liczba_figur
> jeśli :liczba_figur <= 0 [ stop ]
> trybwzoru 2
> wzór    ( zdanie
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           )
> .elipsa ( zdanie
>     ( losowa 400 ) - 200
>     ( losowa 200 ) - 100
>     ( losowa 50 ) + 20
>     ( losowa 30 ) + 20
>           )
> elipsy :liczba_figur - 1
>już
ELIPSY zdefiniowano.
>?cs sż
>?losuj 5 elipsy 10
{narysowanie dziesięciu elips o losowo dobranych współrzędnych środka
oraz średnicach}
```



Rys. 8.28. Rysowanie elips za pomocą procedury .ELIPSA

.GENERATOR .GENER

Składnia: .GENERATOR n

Opis działania: procedura wyboru generatora znaków o numerze n.

Parametr: liczba większa od zera, oznaczająca numer generatora znaków. Generatory znaków są wczytywane przy pomocy poleceń pliku konfiguracyjnego (por. Dodatek F), a nadawane im numery wynikają z kolejności wczytywania. Standardowo jest wczytywany generator znaków z pliku LAT8X14.FNT — ma on zawsze numer 1. Kolejno wczytywane generatory znaków otrzymują numery 2, 3 itd.

Uwagi: jeśli podany parametr przekracza numer ostatniego wczytanego generatora znaków, to procedura nie zmienia aktualnie wybranego generatora znaków i nie sygnalizuje błędu.
Ponieważ wczytywanie generatorów znaków wiąże się z zajmowaniem miejsca w pamięci operacyjnej, należy ograniczyć liczbę wczytywanych generatorów do niezbędnych.
W systemie można jednorazowo wczytać do 16 generatorów znaków.

Składnia: .KOŁO l

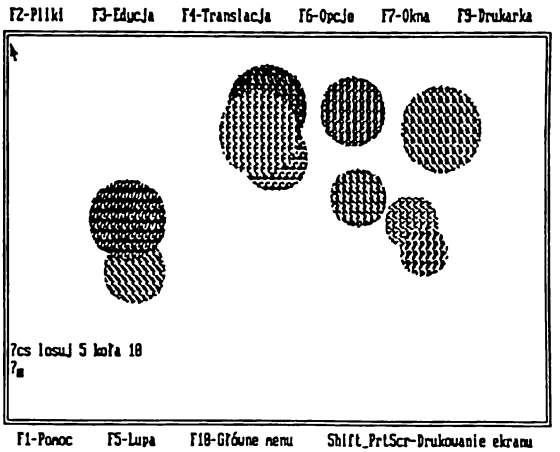
Opis działania: narysowanie koła.

Parametr: lista postaci [x y r], gdzie x i y są liczbami z zakresu (–32000, 32000) oznaczającymi współrzędne środka koła, a r — liczbą z zakresu (0, 32000) oznaczającą jego promień.

Uwagi: sposób wypełniania wnętrza figury zależy od sposobu i wzoru wypełniania, zdefiniowanych za pomocą TRYBWZORU (ang. FILLTYPE) i WZÓR lub WZÓR16 (ang. FILLPATTERN, FILLPATT16).
W zależności od proporcji ekranu ustalonych za pomocą procedury PROPORCJA (ang. SETSCRUNCH), narysowane koło może być mniej lub bardziej "spłaszczone". Dla karty graficznej EGA/VGA najodpowiedniejszą proporcją jest [1 0.775], natomiast dla karty graficznej HGC [1 0.64].

Przykład

```
?oto "koła :liczba_figur
> jeśli :liczba_figur <= 0 [ stop ]
> trybwzoru 2
> wzór    ( zdanie
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           )
> .koło ( zdanie
>     ( losowa 400 ) - 200
>     ( losowa 200 ) - 100
>     ( losowa 30 ) + 20
>           )
> koła :liczba_figur - 1
>już
KOŁA zdefiniowano.
>?cs sż
>?losuj 5 koła 10
{narysowanie dziesięciu kół o losowo dobranych współrzędnych środka
oraz średnicach}
```



Rys. 8.29. Rysowanie kół za pomocą procedury .KOŁO

.KWADRAT **.SQUARE**

Składnia: .KWADRAT l

Opis działania: narysowanie kwadratu.

Parametr: lista postaci [x y a], gdzie x i y są liczbami z zakresu (–32000, 32000) oznaczającymi współrzędne lewego, dolnego rogu kwadratu, a a — liczbą z zakresu (0, 32000) oznaczającą długość jego boku.

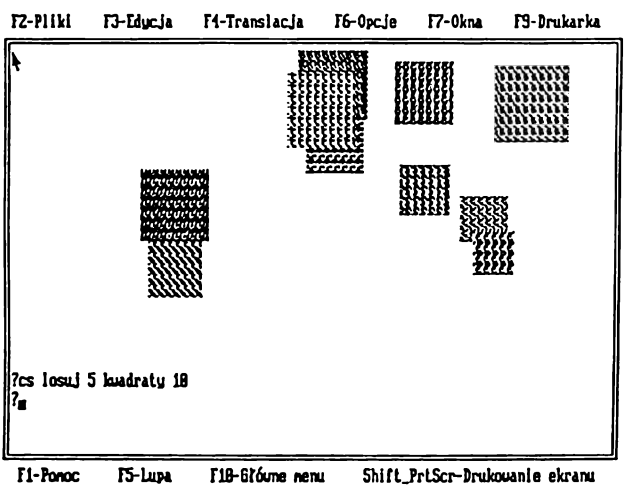
Uwagi: sposób wypełniania wnętrza figury zależy od sposobu i wzoru wypełniania, zdefiniowanych za pomocą TRYBWZORU (ang. FILLTYPE) i WZÓR lub WZÓR16 (ang. FILLPATTERN, FILLPATT16).
W zależności od proporcji ekranu ustalonych za pomocą procedury PROPORCJA (ang. SETSCRUNCH), narysowany kwadrat może być mniej lub bardziej "spłaszczony". Dla karty graficznej EGA/VGA najodpowiedniejszą proporcją jest [1 0.775], natomiast dla karty graficznej HGC [1 0.64].

Przykład

```
?oto "kwadraty :liczba_figur
› jeśli :liczba_figur <= 0 [ stop ]
› trybwzoru 2
```

```
> wzór    ( zdanie
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           )
> .kwadrat ( zdanie
>     ( losowa 400 ) - 200
>     ( losowa 200 ) - 100
>     ( losowa 50 ) + 30
>           )
> kwadraty :liczba_figur - 1
>już
KWADRATY zdefiniowano.
>?cs sż
>?losuj 5 kwadraty 10
```

{narysowanie dziesięciu kwadratów o losowo dobranych współrzędnych oraz bokach}



Rys. 8.30. Rysowanie kwadratów za pomocą procedury .KWADRAT

Składnia: .ŁADUJBIN s n

Opis działania: procedura powodująca załadowanie z dysku do zarezerwowanego obszaru pamięci zawartości pliku o nazwie będącej pierwszym parametrem. Ładowanie następuje począwszy od adresu będącego drugim parametrem. Procedura umożliwia łączenie procedur LOGO z programami napisanymi w innych językach.

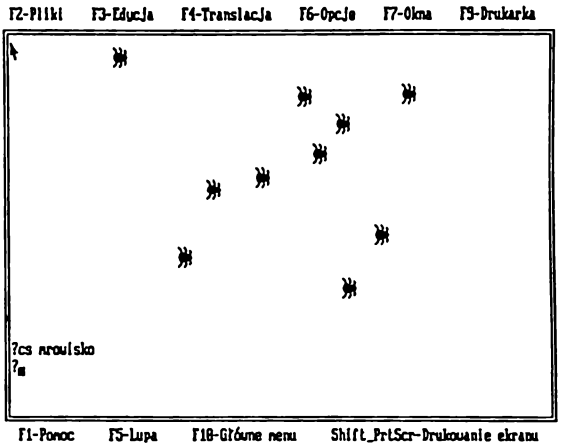
Parametry: pierwszym parametrem jest wyrażenie, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeżeli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie COD. Drugi parametr jest wyrażeniem arytmetycznym o wartości będącej adresem komórki pamięci, od której rozpoczyna się ładowanie. Liczby niecałkowite są zaokrąglane.

Uwaga: procedura .ŁADUJBIN umożliwia załadowanie tylko obszaru pamięci zarezerowanego uprzednio za pomocą procedury .ZAJMIJ (ang. .RESERVE). Dostęp do innych obszarów pamięci, ze względu na dużą możliwość zniszczenia ważnych danych i programu nie jest możliwy. Nie jest także możliwy dostęp do pamięci ekranu (w odróżnieniu od Junior LOGO).

Przykład

```
?oto "mrówka :x :y
> (lokalna "dx "dy "bajt "adres)
> przypisz "adres pierw .zajęte
> przypisz "dy 0
> powtórz 16 {odliczanie współrzędnej Y}
>   [
>     przypisz "dx 0
>     powtórz 2 {odliczanie współrzędnej X}
>       [
>         przypisz "bajt .zobacz :adres
>         przypisz "adres :adres + 1
>         jeśli :bajt > 127
>           [ pkt zdanie :x + :dx :y + :dy
>             przypisz "bajt :bajt - 128 ]
>         przypisz "dx :dx + 1
>         jeśli :bajt > 63
>           [ pkt zdanie :x + :dx :y + :dy
>             przypisz "bajt :bajt - 64 ]
>         przypisz "dx :dx + 1
>         jeśli :bajt > 31
>           [ pkt zdanie :x + :dx :y + :dy
>             przypisz "bajt :bajt - 32 ]
>
```

```
)          przypisz "dx :dx + 1
)          jeśli :bajt ) 15
)            [ pkt zdanie :x + :dx :y + :dy
)              przypisz "bajt :bajt - 16 ]
)          przypisz "dx :dx + 1
)          jeśli :bajt ) 7
)            [ pkt zdanie :x + :dx :y + :dy
)              przypisz "bajt :bajt - 8 ]
)          przypisz "dx :dx + 1
)          jeśli :bajt ) 3
)            [ pkt zdanie :x + :dx :y + :dy
)              przypisz "bajt :bajt - 4 ]
)          przypisz "dx :dx + 1
)          jeśli :bajt ) 1
)            [ pkt zdanie :x + :dx :y + :dy
)              przypisz "bajt :bajt - 2 ]
)          przypisz "dx :dx + 1
)          jeśli :bajt ) 0
)            [ pkt zdanie :x + :dx :y + :dy ]
)          przypisz "dx :dx + 1
)       ]
)     przypisz "dy :dy + 1
)  ]
)już
MRÓWKA zdefiniowano.
?oto "mrówki :liczba_figur
) jeśli :liczba_figur <= 0 [ stop ]
) mrówka ( losowa 400 ) - 200    ( losowa 200 ) - 100
) mrówki :liczba_figur - 1
)już
MRÓWKI zdefiniowano.
?oto "mrowisko
) jeśli ( pierw .zajęte ) = ( ost .zajęte )
)   [
)     .zwolnij
)     .zajmij 300
)     .ladujbin "mrowa.cod pierw .zajęte
)     {załadowanie mapy bitowej wizerunku mrówki z pliku MROWA.COD}
)   ]
)  losuj 2 mrówki 10
)już
MROWISKO zdefiniowano.
?cs mrowisko
```



Rys. 8.31. Postać ekranu po wykonaniu procedury MROWISKO

.ŁAMANA .MULTILINE

Składnia: .ŁAMANA l

Opis działania: narysowanie niedomkniętej linii łamanej

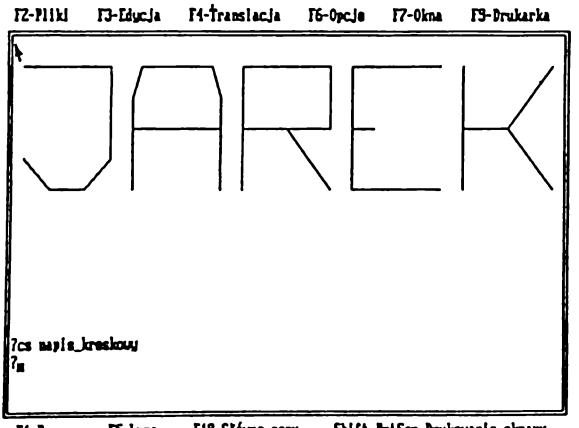
Parametr: lista postaci [n [x y] [x y] ... [x y]], gdzie n jest liczbą wierzchołków łamanej, a następne podlisty są dwuelementowymi listami określającymi współrzędne kolejnych wierzchołków łamanej — są to liczby z zakresu (–32000, 32000). Liczba wierzchołków łamanej n nie może przekraczać 128.

Uwagi: Sposób rysowania poszczególnych odcinków łamanej zależy od sposobu i wzoru kreski, zdefiniowanego za pomocą WZÓRKRESKI.

Przykład

```
?oto "twórz_litery_kreskowe
> przypisz "kreski_J [ 6      [-300 100] [-200 100] [-200  25]
>                             [-230   0] [-270   0] [-300  25] ]
> przypisz "kreski_A [ 9      [-175   0] [-175  75] [-165 100]
>                             [ -85 100] [ -75  75] [ -75  50]
>                             [-175  50] [ -75  50] [ -75   0] ]
```

```
> przypisz "kreski_R [ 7      [ -50   0] [ -50 100] [  50 100]
>                             [  50  50] [ -50  50] [   0  50]
>                             [  50   0] ]
> przypisz "kreski_E [ 7      [ 175   0] [  75   0] [  75  50]
>                             [ 100  50] [  75  50] [  75 100]
>                             [ 175 100] ]
> przypisz "kreski_K [ 7      [ 200 100] [ 200   0] [ 200  50]
>                             [ 250  50] [ 300 100] [ 250  50]
>                             [ 300   0] ]
>już
TWÓRZ_LITERY_KRESKOWE zdefiniowano.
?oto "napis_kreskowy
> .lamana :kreski_J
> .lamana :kreski_A
> .lamana :kreski_R
> .lamana :kreski_E
> .lamana :kreski_K
już
NAPIS_KRESKOWY zdefiniowano.
?twórz_litery_kreskowe    {zdefiniowanie wzorów dla literek J A R E K}
?cs napis_kreskowy        {narysowanie napisu JAREK na ekranie}
```



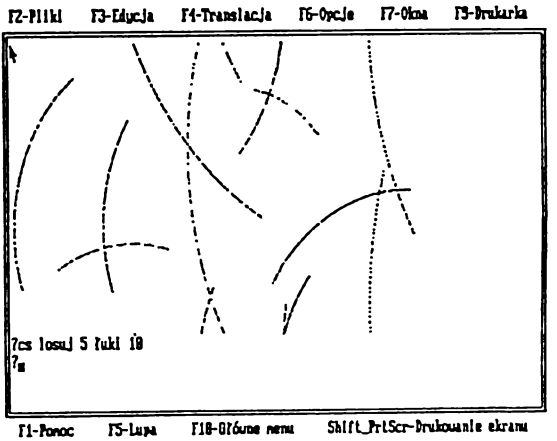
Rys. 8.32. Rysunki wykonane za pomocą procedury .ŁAMANA

Składnia: .ŁUK l

Opis działania: narysowanie łuku.

Parametr: lista postaci [x y r pocz delta], gdzie x i y oznaczają współrzędne środka okręgu, na którego obwodzie jest rysowany łuk, r oznacza promień tego okręgu, pocz oznacza początkowy kąt łuku, a delta oznacza kąt rozwarcia łuku. X, y, pocz, delta są liczbami z zakresu (–32000, 32000), a r jest liczbą z zakresu (0, 32000). Wartości kątów są niejawnie przekształcane do zakresu (0, 359) stopni za pomocą operacji modulo 360.

Uwagi: kąty są liczone tak jak dla żółwia (kąt zerowy oznacza kierunek pionowo w górę).



Rys. 8.33. Rysowanie łuków za pomocą procedury .ŁUK

Przykład

```
?oto "łuki :liczba_figur
> jeśli :liczba_figur <= 0 [ stop ]
> wzórkreski losowa 65535
> .łuk ( zdanie
            ( losowa 400 ) - 200
            ( losowa 200 ) - 100
```

```
            ( losowa 50 ) + 30
            ( losowa 360 )
            ( losowa 360 )
)             )
) łuki :liczba_figur - 1
)już
ŁUKI zdefiniowano.
)?cs sż losuj 5 łuki 10
{narysowanie dziesięciu łuków o losowo dobranych wielkościach}
```

.NUMERBŁĘDU **.ERRORNUMBER**

Składnia: NUMERBŁĘDU → n

Opis działania: funkcja "numer ostatniego błędu", podająca numer błędu, jaki ostatnio spowodował przerwanie wykonywania procedur. Funkcja ta w połączeniu z procedurą .POBŁĘDZIE może być używana do automatycznego wznowienia działania programu po błędzie.

Wynik: numer ostatniego błędu lub 0, gdy błąd nie wystąpił. Numery błędów podano w Dodatku E.

.ODŚMIECANIE **.INSTALL**

Składnia: .ODŚMIECANIE l

Opis działania: procedura ustalająca parametry procedury odzyskującej nieużytki – ODŚMIEĆ (ang. RECYCLE). Procedura odzyskująca nieużytki jest wywoływana w trzech sytuacjach: po jawnym wywołaniu, w przypadku braku pamięci dla nowo tworzonych obiektów oraz automatycznie, co określoną liczbę bajtów przydzielonych zmiennym lub procedurom.

Parametr: trójelementowa lista postaci [n n n] o następującym znaczeniu elementów:

— pierwszy element jest liczbą bajtów z zakresu (1, #FFFF) po przydzieleniu których procedura odzyskująca nieużytki jest automatycznie wywoływana. Standardowo parametr ten jest równy #FFFF, co oznacza, że porządkowanie następuje tylko w sytuacji braku pamięci dla nowo tworzonego obiektu. W większości przypadków zmniejszenie wartości tego parametru powoduje uciążliwe zwiększenie częstości wywołań procedury odzyskującej nieużytki. Jest ono celowe

jedynie w programach operujących na dużej liczbie krótkich list lub słów;

— drugi element jest liczbą określającą czy porządkowanie pamięci ma być sygnalizowane na ekranie komputera. Sygnalizacja polega na wyświetlaniu obracającego się znaku kreski w lewym, górnym rogu ekranu. Jeśli parametr jest równy zeru, to sygnalizacja jest wyłączona. Jeśli parametr jest różny od zera, to sygnalizacja jest włączona Sygnalizacja pozwala stwierdzić, że komputer pracuje poprawnie mimo pozornego zawieszenia działania, na które wskazuje wstrzymanie realizacji programu i zaprzestanie wczytywania znaków z klawiatury w czasie porządkowania pamięci (taki stan może trwać od ułamka sekundy do kilkunastu sekund). Standardowo parametr ma wartość różną od zera, oznaczającą włączenie sygnalizacji,

— trzeci element jest liczbą z zakresu (1, 255) określającą maksymalną liczbę jednorazowo porządkowanych poziomów struktur listowych. Standardowo, parametr ten ma wartość 10. Mniejsza wartość parametru oznacza krótsze, lecz niekompletne odzyskiwanie nieużytków natomiast większa – dłuższe i dokładniejsze. Zmniejszenie parametu jest zatem celowe jedynie dla bardzo prostych programów nie operujących na listach, a zwiększenie jedynie w przypadku programów operujących na listach wielopoziomowych.

Uwaga: dla większości programów wywołanie procedury .ODŚMIECANIE z parametrami pierwszym i trzecim innymi niż standardowe powoduje spowolnienie działania LOGO.

Przykład

```
?.odśmiecanie [1000 0 5]
{Włączenie automatycznego odzyskiwania nieużytków co 1000 bajtów, wyłączenie sygnalizacji i zmniejszenie liczby jednorazowo porządkowanych poziomów list do pięciu}
```

.OKNO? **.WSIZE**

Składnia: .OKNO → l

Opis działania: funkcja podająca rozmiary aktualnego okienka graficznego w punktach ekranu.

Wynik: dwuelementowa lista składająca się z liczb oznaczających szerokość i wysokość okienka graficznego w punktach ekranu.

Przyklad

```
?opisokna [ -50 -50 50 50 ]
{zdefiniowanie okienka graficznego}
?pisz .okno
112 112
{wydrukowanie rozmiarów okienka}
?opisokna [ -100 -50 100 50 ]
?pokaż .okno
[ 208 112 ]
```

.PIERWOTNE **.PRIMITIVES**

Składnia: .PIERWOTNE
lub w analizie wyrażenia
.PIERWOTNE → l

Opis działania: procedura lub funkcja powodująca wyświetlenie nazw wszystkich procedur pierwotnych oraz operatorów LOGO.

Wynik: jeśli funkcja jest wywołana w trakcie analizy wyrażenia, to wynikiem jej działania jest lista zawierająca słowa odpowiadające nazwom wszystkich procedur i funkcji pierwotnych LOGO.

.POBŁĄD **.POERROR**

Składnia: .POBŁĄD n → s

Opis działania: funkcja podająca nazwę procedury użytkownika, która jest wywoływana w momencie wystąpienia błędu o numerze n. W przypadku standardowej obsługi błędu funkcja podaje słowo puste.

Parametr: liczba większa od zera oznaczająca numer błędu.

Wynik: słowo oznaczające nazwę procedury użytkownika uruchamianej w momencie wystąpienia błędu lub słowo puste w przypadku standardowej obsługi błędu.

Uwagi: numery błędów podano w Dodatku E.
W momencie wykonania dowolnej standardowej obsługi błędu (przerwania wykonywania procedur) lub zakończenia wykonywania procedur wszystkie procedury obsługi błędów są ustawiane na standardowe (przerwanie programu i komunikat systemowy o przyczynie błędu).

.POBŁĘDY — .POERRORS

Składnia: .POBŁĘDY

Opis działania: procedura drukująca na ekranie nazwy procedur użytkownika, które są wywoływane w momencie wystąpienia poszczególnych błędów.

Uwagi: numery błędów podano w Dodatku E.
W momencie wykonania dowolnej standardowej obsługi błędu (przerwania wykonywania procedur) lub zakończenia wykonywania procedur wszystkie procedury obsługi błędów są ustawianie na standardowe (przerwanie programu i komunikat systemowy o przyczynie błędu).

.POBŁĘDZIE — .ONERROR

Składnia: .POBŁĘDZIE n s

Opis działania: procedura zmieniająca sposób obsługi błędu o numerze n — po wystąpieniu tego błędu zostanie wywołana procedura o nazwie s.

Parametr: liczba nieujemna oznaczająca numer błędu oraz słowo oznaczające nazwę procedury. Jeśli parametr jest równy 0, to zmiana sposobu obsługi błędu dotyczy wszystkich błędów.

Uwagi: numery błędów podano w Dodatku E.
W momencie wystąpienia błędu są tracone wszelkie informacje o poprzednim wykonywaniu procedur (ślady powrotu do procedur wywołujących). Oznacza to, że ewentualny powrót z procedury s kończy wykonywanie procedur i powoduje powrót do edytora wierszowego.
Można zmienić także sposób normalnego końca pracy programu (błąd numer 1) — w ten sposób uniemożliwia się wyjście do edytora wierszowego.
W momencie wykonania dowolnej standardowej obsługi błędu (przerwania wykonywania procedur) lub zakończenia wykonywania procedur wszystkie procedury obsługi błędów są ustawianie na standardowe (przerwanie programu i komunikat systemowy o przyczynie błędu).
Sposobu obsługi niektórych błędów nie można zmienić. Dotyczy to tak zwanych błędów "fatalnych" — ich opis zawarto w Dodatku E.

Przykład:

```
?oto "dzielenie_przez_zero
> .poblędzie 16 "błąd
> {Błąd numer 16 to próba dzielenia przez zero}
>  ps 1 / 0
>  ps [ Z powrotem w procedurze DZIELENIE_PRZEZ_ZERO. ]
>  ps [ Te napisy nie powinny być nigdy wydrukowane. ]
>już
DZIELENIE_PRZEZ_ZERO zdefiniowano.
?oto "błąd
>  ps [Wykryto próbę dzielenia przez zero !!!!!]
>  {powrót do edytora wierszowego}
>już
BŁĄD zdefiniowano.
?dzielenie_przez_zero
Wykryto próbę dzielenia przez zero !!!!!
{Po przerwaniu pracy programu i powrocie do edytora wierszowego błąd
"dzielenie przez zero" jest obsługiwany w sposób standardowy}
?ps 1 / 0
Próba dzielenia przez zero.
```

.PROSTOKĄT — .RECTANGLE

Składnia: .PROSTOKĄT l

Opis działania: narysowanie prostokąta.

Parametr: lista postaci [x y a b], gdzie x i y są liczbami z zakresu (–32000, 32000) oznaczającymi współrzędne lewego, dolnego rogu prostokąta, a a i b — liczbami z zakresu (0, 32000) oznaczającą długość jego boków.

Parametr: lista postaci [x y a b], gdzie poszczególne elementy są liczbami z zakresu (–32000, 32000) oznaczającymi odpowiednio współrzędne lewego, dolnego rogu prostokąta (x,y) oraz długości jego boków a i b.

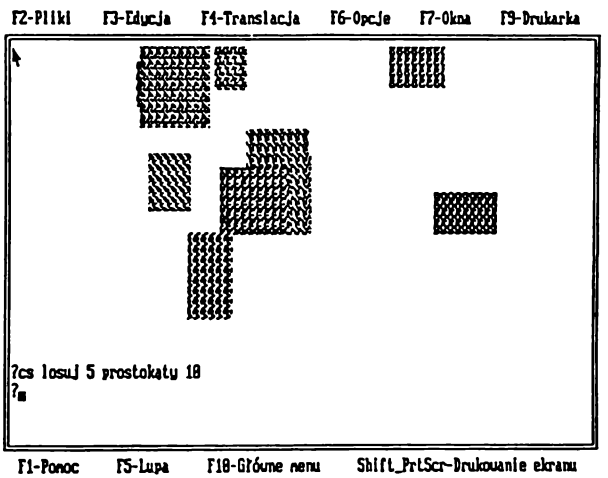
Uwagi: sposób wypełniania wnętrza figury zależy od sposobu i wzoru wypełniania, zdefiniowanych za pomocą TRYBWZORU (ang. FILLTYPE) i WZÓR lub WZÓR16 (ang. FILLPATTERN, FILLPATT16).

Przykład

```
?oto "prostokąty :liczba_figur
> jeśli :liczba_figur <= 0 [ stop ]
> trybwzoru 2
```

```
> wzór      ( zdanie
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           losowa 255  losowa 255
>           )
> .prostokąt ( zdanie
>    ( losowa 400 ) - 200
>    ( losowa 200 ) - 100
>    ( losowa 50 ) + 30
>    ( losowa 30 ) + 30
>           )
> prostokąty :liczba_figur - 1
>już
PROSTOKĄTY zdefiniowano.
>?cs sż
>?losuj 5 prostokąty 10
```

{narysowanie dziesięciu prostokątów o losowo dobranych współrzędnych oraz wielkościach}



Rys. 8.34. Rysowanie prostokątów za pomocą procedury .PROSTOKĄT

.RS232 — .SERIALMODE

Składnia: .RS232 l

Opis działania: procedura ustalająca parametry komunikacji przez złącze transmisji szeregowej RS–232C.

Parametr: sześcioelementowa lista postaci [n l sp s c p], gdzie poszczególne parametry mają następujące znaczenie: n — słowo oznaczające nazwę urządzenia (w systemie MS–DOS), do którego jest dołączony port transmisji szeregowej, l — numer portu złącza z zakresu 1 do 4, sp — szybkość transmisji w bitach na sekundę (maksymalnie 9600 bitów/sek), s — liczba bitów stopu (1 lub 2), c — liczba bitów znaku (7 lub 8), p — sposób kontroli błędu parzystości (0–bez kontroli parzystości, 1 – kontrola "parzysta", 2 – kontrola "nieparzysta").

Uwagi: wybranie niewłaściwych parametrów może spowodować trudne do przewidzenia błędy, na przykład pomyłka w nazwie urządzenia może spowodować błędy w pracy myszki (dołączonej standardowo do portu o nazwie COM1).

Przykłady:

```
{zdefiniowanie portu COM2 — szybkość 9600 bitów/sek, dwa bity stopu,
8 bitów danych, brak kontroli parzystości}
?.RS232 [ com2 2 9600 2 8 0 ]
```

.RS232? — .SERIALACTIVE (.SA)

Składnia: .RS232? → b

Opis działania: funkcja o wartości PRAWDA, gdy jest gotowy do odebrania znak ze złącza transmisji szeregowej RS–232C i FAŁSZ w przeciwnym razie.

Wynik: wartość logiczna PRAWDA lub FAŁSZ.

Przykłady:

```
oto "czekaj_na_znak
› jeśli .rs232
›    [ ps [Znak gotowy do odebrania.] stop ]
› czekaj_na_znak
już
CZEKAJ_NA_ZNAK zdefiniowano.
?czekaj_na_znak
{oczekiwanie na gotowość odebrania znaku z RS–232C}
Znak gotowy do odebrania.
?
```

.SYSTEM .SYSTEM

Składnia: .SYSTEM

Opis działania: zakończenie pracy w LOGO i przejście do systemu operacyjnego MS–DOS.

Uwagi: obejrzyj przykład dla procedury DOŚĆ (ang. BYE).

Przykład

```
?oto wyjście
> pisz [Kończymy pracę w LOGO]
> przypisz "wyjście cz    {oczekiwanie na dowolny znak}
> .system
>już
WYJŚCIE zdefiniowano
?wyjście
Kończymy pracę w LOGO
```

.SZYBKOŚĆ .SPEED

Składnia: .SZYBKOŚĆ n

Opis działania: procedura zmiany szybkości wykonywania procedur przez dodanie po wykonaniu każdej procedury pierwotnej opóźnienia n milisekund.

Parametr: liczba oznaczająca opóźnienie wykonywania programu w milisekundach z zakresu (0, 65535). Jeśli parametr jest, równy 65535, to program jest wykonywany krokowo — po wykonaniu każdej procedury pierwotnej na ekranie pojawia się okienko "praca krokowa" i wykonywanie dalszej części programu jest zatrzymywane aż do czasu naciśnięcia dowolnego klawisza.

Przykład

```
?oto "pułapka_na_żółwia
> sż grafika
> podnieś poz [–123 0] opuść
> naprzód 83 prawo 90 naprzód 246 prawo 90 naprzód 166 prawo 90
> naprzód 246 prawo 90 naprzód 83
> podnieś wróć
> kąt losowa 90
> podnieś pż
> przypisz "opóźnienie 0
> ruch_żółwia
>już
```

```
PUŁAPKA_NA_ŻÓŁWIA zdefiniowano.
?oto "ruch_żółwia
> jeśli (abs pierw poz?) > 100 [ wstecz 10 prawo losowa 90 ]
> jeśli (abs ost poz?) > 60 [ wstecz 10 prawo losowa 90 ]
> naprzód 10
> jeśli klawisz?
>   [przypisz "klawisz cz
>    jeśli :klawisz = "\+ [ przypisz "opóźnienie :opóźnienie + 10 ]
>    jeśli :klawisz = "\— [ przypisz "opóźnienie :opóźnienie — 10 ]
>    jeśli :opóźnienie < 0 [przypisz "opóźnienie 0]
>  ]
> .szybkość :opóźnienie
> ruch_żółwia
>już
RUCH_ŻÓŁWIA zdefiniowano.
?pułapka_na__żółwia
{żółw na ekranie porusza się w obrębie narysowanego prostokąta; szybkość
poruszanie się żółwia można zmieniać za pomocą klawiszy "+" i "—"}
```

.TRÓJKĄT **.TRIANGLE**

Składnia: .TRÓJKĄT l

Opis działania: narysowanie trójkąta.

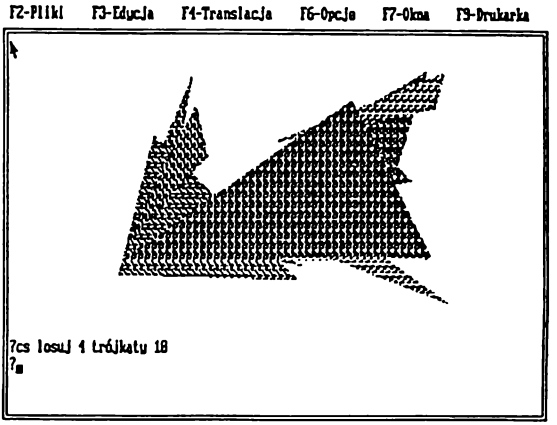
Parametr: lista postaci [[x1 y1] [x2 y2] [x3 y3]], gdzie poszczególne elementy są dwuelementowymi listami składającymi się z liczb z zakresu (—32000, 32000), oznaczającymi odpowiednio współrzędne trzech wierzchołków trójkąta.

Uwagi: sposób wypełniania wnętrza figury zależy od sposobu i wzoru wypełniania, zdefiniowanych za pomocą TRYBWZORU (ang. FILLTYPE) i WZÓR lub WZÓR16 (ang. FILLPATTERN, FILLPATT16).

Przykład

```
?oto "trójkąty :liczba_figur
  (lokalna "xy1 "xy2 "xy3)
> jeśli :liczba_figur <= 0 [ stop ]
> trybwzoru 2
> wzór   ( zdanie
>          losowa 255  losowa 255
>          losowa 255  losowa 255
>          losowa 255  losowa 255
>          losowa 255  losowa 255
>          )
```

```
) przypisz "xy1 zdanie ( losowa 400 ) – 200  ( losowa 200 ) – 100
) przypisz "xy1 zdanie ( losowa 400 ) – 200  ( losowa 200 ) – 100
) przypisz "xy1 zdanie ( losowa 400 ) – 200  ( losowa 200 ) – 100
) .trójkąt ( lista :xy1 :xy2 :xy3 )
) trójkąty :liczba_figur – 1
)już
TRÓJKĄTY zdefiniowano.
)?cs sż
)?losuj 5 trójkąty 10
```
{narysowanie dziesięciu trójkątów o losowo dobranych współrzędnych wierzchołków}



Rys. 8.35. Rysowanie trójkątów za pomocą procedury .TRÓJKĄT

.UMIEŚĆ **.DEPOSIT**

Składnia: .UMIEŚĆ n n

Opis działania: procedura powodująca zapisanie w zarezerowanym obszarze pamięci pod adresem określonym przez pierwszy parametr bajtu określonego przez drugi parametr.

Parametry: pierwszym parametrem jest wyrażenie arytmetyczne z zakresu (0, 65535). Drugi parametr jest wyrażeniem arytmetycznym z zakresu (0, 255). Liczby niecalkowite są zaokrąglane.

Uwaga: Wywołanie procedury .UMIEŚĆ z adresem wykraczającym poza obszar zarezerwowany za pomocą procedury .ZAJMIJ (ang. RESERVE) powoduje błąd.

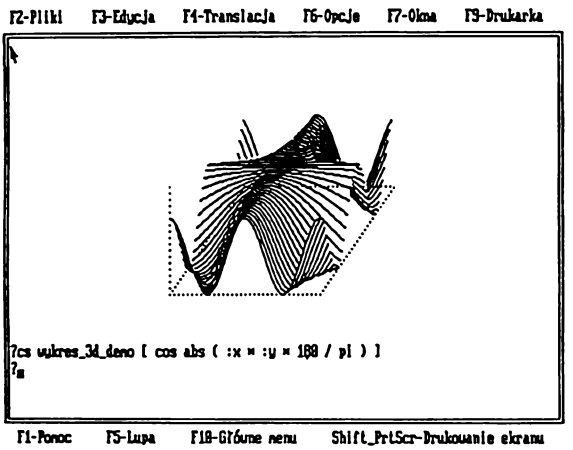
Przykład

{implementacja struktur tablicowych w pamięci zarezerowanej przy pomocy procedury .ZAJMIJ; sposób ten jest bardzo efektywny dla tablic liczb z zakresu od 0 do 255 (tablic bajtów)}

```
?oto "wykres_3d
> lokalna "warunek
> przypisz "i 1
> przypisz "sx ( :sx – :zx ) / 170
> przypisz "sy ( :sy – :zy ) / 85
> przypisz "sz 90 / ( :sz – :zz )
> przypisz "d prawda
> przypisz "n –85
> powtórz 170 / 5
>   [
>     pkt lista :n – 42 –80
>     pkt lista :n + 43 5
>     przypisz "n :n + 5
>   ]
> przypisz "n –80
> powtórz 90 / 5
>   [
>     pkt lista –127 :n
>     przypisz "n :n + 5
>   ]
> przypisz "n –80
> powtórz 1 + ( 85 / 3 )
>   [
>     pkt lista :n + 123 :n
>     pkt lista :n – 47 :n
>     przypisz "n :n + 3
>   ]
> przypisz "m 0
> powtórz 42
>   [
>     przypisz "n –1
>     podnieś
>     powtórz 85
>       [
>         przypisz "n :n + 2
>         przypisz "y :m * :sy + :zy
>         przypisz "x :n * :sx + :zx
>         przypisz "i :n + :m
```

```
>          przypisz "z ( ( zrób :funkcja ) - :zz ) * :sz + :m
>          przypisz "warunek falsz
>          {sprawdzenie zakresu dla rysowanego punktu}
>          jeśli ( (.pokaż :minimum + :i) > :z )
>           [
>            przypisz "warunek prawda
>            .umieść :minimum + :i :z
>           ]
>          jeśli ( (.pokaż :maksimum + :i) < :z )
>           [
>            przypisz "warunek prawda
>            .umieść :maksimum + :i   :z
>           ]
>          jeśli :warunek
>           [
>            poz lista :i - 127 :z - 80 opuść
>           ]
>          inaczej
>           [ podnieś ]
>         ]
>       przypisz "m :m + 2 pu
>     ]
>już
WYKRES_3D zdefiniowano.
?oto "wykres_3d_demo :fun
>  lokalna "i
>  jeśli (pierw .zajęte ) = ost .zajęte
>   [
>    .zwolnij
>    .zajmij 1024
>    przypisz "adres pierw .zajęte
>    {zarezerwowanie miejsca na dwie tablice po 512 bajtów}
>    {tablice te pamiętają wartości minimalne i maksymalne funkcji
>     dla każdej współrzędnej X (uwzględniając przesunięcie
>     wynikające z dodania trzeciego wymiaru)}
>   ]
>  przypisz "i 0
>  przypisz "minimum :adres
>  przypisz "maksimum :adres + 512
>  powtórz 256 [ .umieść :minimum + :i 255 przypisz "i :i + 1 ]
>  {zainicjowanie tablicy wartości minimalnych funkcji}
>  przypisz "i 0
>  powtórz 256 [ .umieść :maksimum + :i 0   przypisz "i :i + 1 ]
>  {zainicjowanie tablicy wartości maksymalnych funkcji}
```

```
> przypisz "funkcja :fun
> przypisz "zx -2.5
> przypisz "sx 2.5
> przypisz "zy -2.5
> przypisz "sy 2.5
> przypisz "zz -1
> przypisz "sz 2
> wykres_3d
>już
WYKRES_3D_DEMO zdefiniowano.
?wykres_3d_demo [ cos abs ( :x * :y * 180 / pi ) ]
```



Rys. 8.36. Rysunek uzyskany za pomocą procedury WYKRES_3D

.UMIEŚĆP **.OUT**

Składnia: .UMIEŚĆP I

Opis działania: procedura powodująca wysłanie podanego bajtu do portu wejścia/wyjścia o podanym adresie.

Parametry: pierwszym parametrem jest wyrażenie arytmetyczne o wartości z zakresu od 0 do 65535 będącej adresem portu wejścia/wyjścia. Drugi parametr jest wyrażeniem arytmetycznym o wartości z zakresu (0, 255) będącej wysyłanym bajtem. Liczby niecalkowite są zaokrąglane.

Uwaga: procedury .UMIEŚĆP należy używać ze szczególną rozwagą, ponieważ podanie niewłaściwego adresu może spowodować trudne do wykrycia błędy, a nawet utratę kontroli nad działaniem komputera.

.WIELOKĄT .POLYGON

Składnia: .WIELOKĄT l

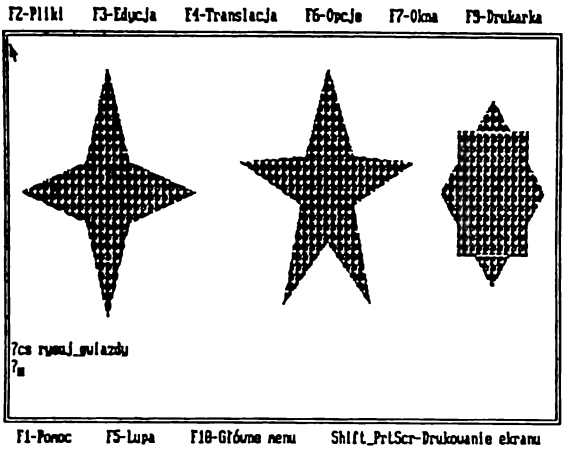
Opis działania: narysowanie wielokąta (domkniętej linii łamanej).

Parametr: lista postaci [n [x y] [x y] ... [x y]], gdzie pierwszy element oznacza liczbę wierzchołków wielokąta, a następne są dwuelementowymi listami składającymi się z liczb z zakresu (–32000, 32000), oznaczającymi współrzędne kolejnych wierzchołków wielokąta. Liczba wierzchołków n nie może przekraczać 128.

Uwagi:. sposób wypełniania wnętrza figury zależy od sposobu i wzoru wypełniania, zdefiniowanych za pomocą TRYBWZORU (ang. FILLTYPE) i WZÓR lub WZÓR16 (ang. FILLPATTERN, FILLPATT16).

Przykład

```
?oto "twórz_gwiazdy
> przypisz "gwiazda4 [ 8      [–300 0]   [–225 25]   [–200 100]
>                            [–175 25]  [–100 0]
>                            [–175 –25] [–200 –100] [–225 –25] ]
> przypisz "gwiazda5 [ 10     [–50 25]    [ 25 30]    [50 100]
>                            [80 30]    [ 150 25]    [80 –10]
>                            [100 –90]   [ 50 –40]    [0 –90]
>                            [20 –10] ]
> przypisz "ośmiokąt [ 16     [180 0]    [200 25]    [200 50]
>                            [220 50]    [240 75]    [ 260 50]
>                            [280 50]    [280 25]    [300 0]
>                            [280 –25]   [280 –50]    [260 –50]
>                            [240 –75]   [220 –50]    [200 –50]
>                            [200 –25] ]
już
TWÓRZ_GWIAZDY zdefiniowano.
?oto "rysuj_gwiazdy
> .wielokąt :gwiazda4
> .wielokąt :gwiazda5
> .wielokąt :ośmiokąt
>już
RYSUJ_GWIAZDY zdefiniowano.
?twórz_gwiazdy
?cs rysuj_gwiazdy
```



Rys. 8.37. Rysunki uzyskane za pomocą procedury .WIELOKĄT

.WYCINEK .PIESLICE

Składnia: .WYCINEK l

Opis działania: narysowanie wycinka koła.

Parametr: lista postaci [x y r pocz delta], gdzie x i y oznaczają współrzędne środka okręgu, na którego bazie jest rysowany łuk, r oznacza promień tego okręgu, pocz oznacza początkowy kąt wycinka, a delta — jego kąt rozwarcia. X, y, pocz, delta są liczbami z zakresu (−32000, 32000), a r jest liczbą z zakresu (0, 32000). Wartości kątów są niejawnie przekształcane do zakresu (0,6 359) stopni za pomocą operacji modulo 360.

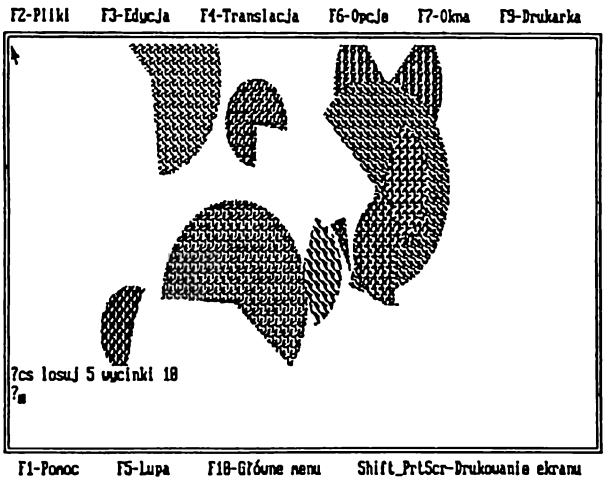
Uwagi: kąty są liczone tak jak dla żółwia (kąt zerowy oznacza kierunek pionowo w górę).
Sposób wypełniania wnętrza figury zależy od sposobu i wzoru wypełniania, zdefiniowanych za pomocą TRYBWZORU (ang. FILLTYPE) i WZÓR lub WZÓR16 (ang. FILLPATTERN, FILLPATT16).

Przykład

```
?oto "wycinki_kół :liczba_figur
```

```
> jeśli :liczba_figur <= 0 [ stop ]
> trybwzoru 2
> wzór    ( zdanie
>          losowa 255 losowa 255
>          losowa 255 losowa 255
>          losowa 255 losowa 255
>          losowa 255 losowa 255
>          )
> .wycinek ( zdanie
>     ( losowa 400 ) - 200
>     ( losowa 200 ) - 100
>     ( losowa 50 ) + 30
>     ( losowa 360 )
>     ( losowa 360 )
>          )
> wycinki_kół :liczba_figur - 1
>już
WYCINKI_KÓŁ zdefiniowano.
>?cs sż
>?losuj 5 wycinki_kół 10
```

{narysowanie dziesięciu wycinków kół o losowo dobranych współrzędnych środka oraz średnicach i kątach}



Rys. 8.38. Rysowanie wycinków kół za pomocą procedury .WYCINEK

.WYWOŁAJ **.CALL**

Składnia: .WYWOŁAJ n → n

Opis działania: funkcja, której wartością jest zawartość rejestru AX mikroprocesora po wykonaniu procedury zaczynającej się od adresu będącego parametrem, znajdującej się w zarezerwowanym obszarze pamięci.

Parametr: wyrażenie arytmetyczne o wartości z zakresu od (0, 65535), będącej adresem procedury w języku asemblera. Liczby niecałkowite są zaokrąglane.

Wynik: liczba z zakresu od 0 do 65535 będąca zawartością rejestru AX mikroprocesora, po wykonaniu procedury.

Uwaga: procedury .WYWOŁAJ należy używać ze szczególną rozwagą, ponieważ podanie niewłaściwego adresu powoduje zazwyczaj trudne do wykrycia błędy, a nawet utratę kontroli nad działaniem komputera.
Wywołanie procedury .WYWOŁAJ z adresem wykraczającym poza obszar zarezerwowany za pomocą procedury .ZAJMIJ (ang. RESERVE) powoduje błąd.

.ZAJĘTE **.RESERVED**

Składnia: .ZAJĘTE → l

Opis działania: funkcja, której wartością jest lista określająca rozmiar pamięci zarezerwowanej dla programów w językach innych niż LOGO.

Wynik: dwuelementowa lista postaci [n n]. Pierwszym elementem jest adres pierwszego bajtu zarezerwowanego obszaru pamięci, a drugim — adres pierwszego bajtu bezpośrednio za zarezerwowanym obszarem pamięci.

Przykład

```
?pisz .zajęte
0 0
?.zajmij 100
?pokaż .zajęte
[ 0 100 ]
```

.ZAJMIJ	.RESERVE
Składnia:	.ZAJMIJ n
Opis działania:	procedura powodująca zarezerwowanie obszaru pamięci, o rozmiarze będącym parametrem, na programy w językach innych niż LOGO.
Parametr:	wyrażenie arytmetyczne, którego wartość jest rozmiarem rezerwowanego obszaru pamięci w bajtach.
Uwaga:	suma wszystkich zarezerwowanych obszarów nie może przekroczyć 65535 bajtów (64 kB).

.ZAPISZBIN	.BSAVE
Składnia:	.ZAPISZBIN s l
Opis działania:	procedura powodująca przesłanie do pliku o nazwie określonej przez pierwszy parametr, zawartości zarezerowanego obszaru pamięci o rozmiarze określonym przez drugi parametr.
Parametry:	pierwszy parametr jest wyrażeniem, którego wartością jest słowo będące jednoznaczną nazwą pliku. Jeśli podana nazwa nie zawiera rozszerzenia, to dodawane jest rozszerzenie .COD. Drugi parametr jest wyrażeniem, którego wartością jest dwuelementowa lista postaci [n n]. Oba jej elementy są liczbami z zakresu (0, 65535). Pierwszy element jest adresem początku obszaru w zarezerwowanym obszarze pamięci, a drugi jego rozmiarem w bajtach.

.ZOBACZ	.EXAMINE
Składnia:	.ZOBACZ n → n
Opis działania:	funkcja podająca zawartość komórki pamięci z zarezerwowanego obszaru pamięci o podanym adresie.
Parametr:	wyrażenie arytmetyczne o wartości z zakresu (0, 65535) będącej adresem komórki pamięci. Liczby niecałkowite są zaokrąglane.
Wynik:	liczba całkowita z zakresu (0, 255). Wywołanie procedury .ZOBACZ z adresem wykraczającym poza obszar zarezerwowany za pomocą procedury .ZAJMIJ (ang. RESERVE) powoduje błąd.

.ZOBACZP .IN

Składnia: .ZOBACZP n → n

Opis działania: funkcja, której wartością jest zawartość portu wejścia/wyjścia o adresie określonym przez parametr.

Parametr: wyrażenie arytmetyczne o wartości z zakresu od 0 do 65535. Liczby niecałkowite są zaokrąglane.

Wynik: liczba całkowita z zakresu od 0 do 255.

.ZWOLNIJ .FREE

Składnia: .ZWOLNIJ

Opis działania: procedura powodująca zwolnienie pamięci zajętej uprzednio przez procedurę .ZAJMIJ.

Przykład

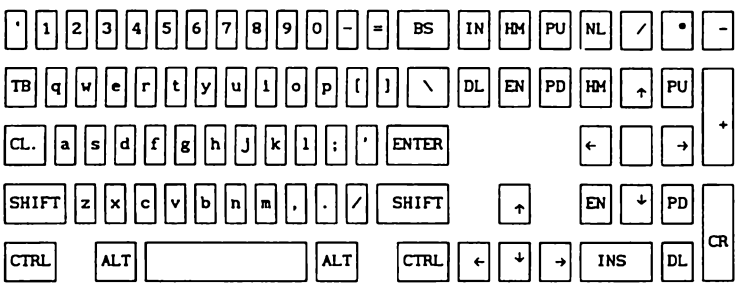
```
?.zajmij 100
?pisz .zajęte
 0 100
?.zwolnij
?pisz .zajęte
 0 0
```

Dodatek A.
ROZSZERZONY KOD ASCII LATIN-II

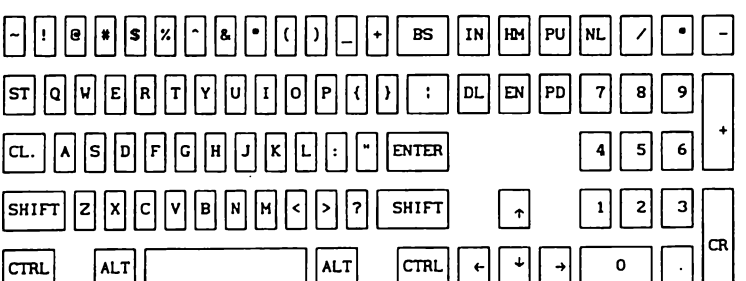
	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	NUL							BEL	BS		LF			CR		
10																
20	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	:	}	~	DEL
80	C	u	e	a	a	u	ć	c	ł	e	O	o	i	Ź	A	Ć
90	E	L	l	o	O	L	l	Ś	ś	O	U	Y	t	Ł	α	c
A0	a	i	ó	u	Ą	ą	Ź	z	Ę	ę	λ	ź	C	s	«	»
B0	▒	▓	▓	│	┤	A	A	E	S	╣	▌	╗	╝	Ż	ż	┐
C0	└	┴	┬	├	─	┼	A	a	╚	╔	╩	╦	╠	=	╬	
D0	d	D	D	E	d	N	I	I	e	┘	┌	▓	▄	T	U	■
E0	Ó	β	O	Ń	ń	n	S	s	R	U	r	U	y	Y	t	'
F0	Ω	"	'			ƒ	+	'	•	"	'	u	R	r		²

Dodatek B.
ROZMIESZCZENIE ZNAKÓW NA KLAWIATURZE

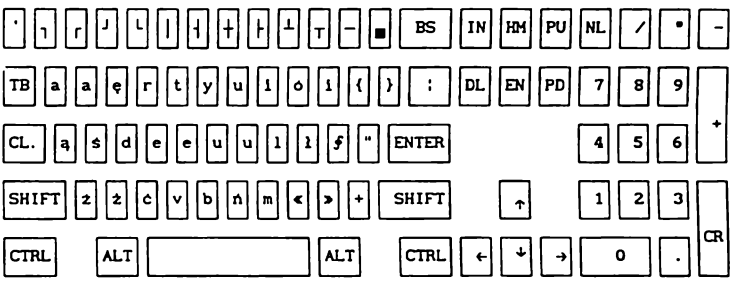
Klawiatura podstawowa



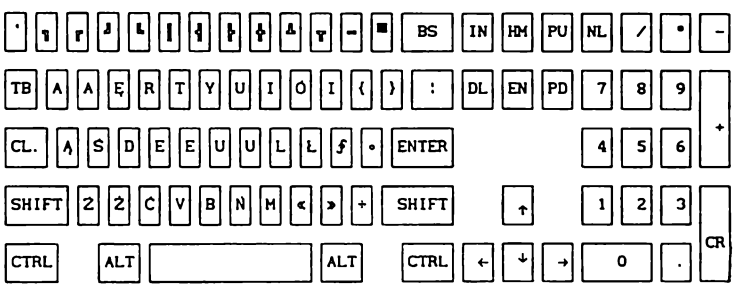
Klawiatura z wciśniętym klawiszem SHIFT



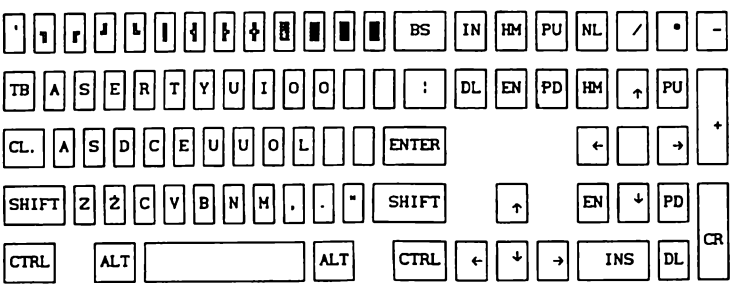
Klawiatura z wciśniętym klawiszem ALT



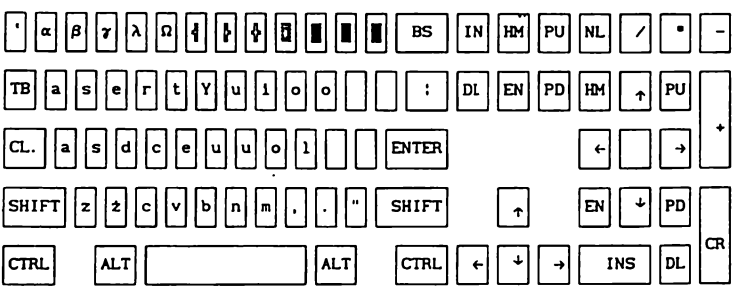
Klawiatura z wciśniętymi klawiszami ALT i SHIFT



Klawiatura z wciśniętymi klawiszami ALT, SHIFT i CTRL



Klawiatura z wciśniętymi klawiszami ALT i CTRL



Dodatek C.
SŁOWNICZEK POLSKO-ANGIELSKI NAZW PROCEDUR PIERWOTNYCH

Dodatek ten zawiera spis polskich nazw procedur pierwotnych języka AC-LOGO wraz z ich angielskimi odpowiednikami. W przypadku procedur, których nazwy mają skrót, alternatywne nazwy podano w nawiasach.

ABS	—	ABS
ARCCOS	—	ARCCOS
ARCSIN	—	ARCSIN
ARCTG	—	ARCTAN
ASCII	—	ASCII
AZYMUT	—	TOWARDS
BEZOST (BO)	—	BUTLAST (BL)
BEZPIERW (BP)	—	BUTFIRST (BF)
BO (BEZOST)	—	BUTLAST (BL)
BP (BEZPIERW)	—	BUTFIRST (BF)
CL (CZYTAJLISTĘ)	—	READLIST (RL)
COS	—	COS
CRS (CZYTAJRS232)	—	READSERIALCHAR (RSC)
CS (CZYŚĆ)	—	CLEARSCREEN (CS)
CZ (CZYTAJZNAK)	—	READCHAR (RC)
CZEKAJ	—	WAIT
CZYŚĆ (CS)	—	CLEARSCREEN (CS)
CZYTAJLISTĘ (CL)	—	READLIST (RL)
CZYTAJRS232 (CRS)	—	READSERIALCHAR (RSC)
CZYTAJZNAK (CZ)	—	READCHAR (RC)
DŁUGOŚĆ	—	COUNT
DOŚĆ	—	BYE
DRUKUJOBRAZ	—	COPYSCREEN (CPS)

DRUKUJOKNO	—	COPYWINDOW (CPW)
DYSK	—	SETDRIVE
DYSK?	—	DRIVE
DZIELEKRAN	—	SETSPLIT
ELEMENT	—	ITEM
ELEMENT?	—	MEMBERP
ENT (ENTIER)	—	INT
ENTIER (ENT)	—	INT
EXP	—	EXP
FAŁSZ	—	FALSE
GR (GRAFIKA)	—	FULLSCREEN (FS)
GRAFIKA (GR)	—	FULLSCREEN (FS)
I	—	AND
ILOCZYN	—	PRODUCT
ILORAZC	—	QUOTIENT
INACZEJ	—	ELSE
JASKRAWO	—	BRIGHTON (BON)
JEŚLI	—	IF
JUŻ	—	END
KAT (KATALOG)	—	CAT
KATALOG (KAT)	—	CAT
KĄT	—	SETHEADING (SETH)
KĄT?	—	HEADING
KLAWISZ?	—	KEYPRES (KP)
KOLORP?	—	POINTCOLOR
KOLORTEKSTU (KT)	—	TEXTCOLOR (TC)
KT (KOLORTEKSTU)	—	TEXTCOLOR (TC)
KURSOR	—	SETCURSOR
KURSOR?	—	CURSOR
LEWO (LW)	—	LEFT (LT)
LEWYPRZYCISK (LPM)	—	MOUSELEFTBUTTON (MLB)
LICZBA?	—	NUMBERP
LISTA	—	LIST
LISTA?	—	LISTP
LN	—	LN
LOKALNA	—	LOCAL
LOSOWA	—	RANDOM
LOSUJ	—	RERANDOM
LPM (LEWYPRZYCISK)	—	MOUSELEFTBUTTON (MLB)

LUB	—	OR
LW (LEWO)	—	LEFT (LT)
ŁADUJ	—	LOAD
ŁADUJO	—	LOADSCR
ŁADUJOKNO	—	LOADWND
ŁADUJRED	—	LOADED
ŁADUJW	—	LOADOBJ
MARGINESYDRUKU	—	SETCOPYMARGIN
MIGAJĄCO	—	FLASHON (FON)
ML	—	LOWERCASE (LC)
NAK	—	LPUT
NAP	—	FPUT
NAPRZÓD (NP)	—	FORWARD (FD)
NIE	—	NOT
NIEJASKRAWO	—	BRIGHTOFF (BOF)
NIEMIGAJĄCO	—	FLASHOFF (FOF)
NP (NAPRZÓD)	—	FORWARD (FD)
NUTA	—	SOUND
ODLICZANIE	—	STARTTIMER
ODŚMIEĆ	—	RECYCLE
ODWR (ODWRACANIE)	—	PENREVERSE (PX)
ODWRACANIE (ODWR)	—	PENREVERSE (PX)
OKNO	—	WINDOW
OKREŚL	—	DEFINE
OKREŚLONE?	—	DEFINEDP
OPISOKNA	—	DEFINEWINDOW (DFW)
OPÓŹNIENIE	—	DELAY
OPU (OPUŚĆ)	—	PENDOWN (PD)
OPUŚĆ (OPU)	—	PENDOWN (PD)
ORIENTUJDRUK	—	SETCOPYORIENTATION (SCO)
OST	—	LAST
OTO	—	TO
PI	—	PI
PIERW	—	FIRST
PIERWIASTEK (PWK)	—	SQRT
PIERWOTNE?	—	PRIMITIVEP
PISAK	—	PENCOLOR (PC)
PISZ (PS)	—	PRINT (PR)
PISZRS232	—	SERIALOUT

PKT	—	DOT
PLIKI	—	FCAT
PŁOT	—	FENCE
PM	—	SHOWMOUSE (SM)
PO	—	PRINTOUT (PO)
POD (PODNIEŚ)	—	PENUP (PU)
PODNIEŚ (POD)	—	PENUP (PU)
POKAŻ	—	SHOW
PONP	—	POTS
POW	—	POALL
POWP	—	POPS
POWTÓRZ	—	REPEAT
POWZ	—	PONS
POZ	—	SETXY
POZ?	—	POSITION (POS)
POZX	—	SETX
POZX?	—	XCOR
POZY	—	SETY
POZY?	—	YCOR
PPM (PRAWYPRZYCISK)	—	MOUSERIGHTBUTTON (MRB)
PRAWDA	—	TRUE
PRAWO (PW)	—	RIGHT (RT)
PRAWYPRZYCISK (PPM)	—	MOUSERIGHTBUTTON (MRB)
PROPORCJA	—	SETSCRUNCH
PROPORCJA?	—	SCRUNCH
PRZERWIJ	—	TOPLEVEL
PRZYP (PRZYPISZ)	—	MAKE
PRZYPISZ (PRZYP)	—	MAKE
PS (PISZ)	—	PRINT (PR)
PUŃKT?	—	POINTP
PUSTE?	—	EMPTYP
PW (PRAWO)	—	RIGHT (RT)
PWK (PIERWIASTEK)	—	SQRT
PŻ	—	SHOWTURTLE (ST)
RAMKA	—	BORDERCOLOR (BC)
RED	—	EDIT (ED)
REDW	—	EDALL
RESZTA	—	REMAINDER
RÓWNE?	—	EQUALP

SIGN	—	SIGN
SIN	—	SIN
SKALUJDRUK	—	SETCOPYSCALE (SCS)
SKLEJ	—	WRAP
SŁOWO	—	WORD
SŁOWO?	—	WORDP
SM	—	HIDEMOUSE (HM)
SORTUJ	—	SORT
STOP	—	STOP
STOPER	—	TIMER
SUMA	—	SUM
SŻ	—	HIDETURTLE (HT)
ŚCIER (ŚCIERANIE)	—	PENERASE (PE)
ŚCIERANIE (ŚCIER)	—	PENERASE (PE)
ŚCIEŻKA	—	PATH
ŚPM (ŚRODKOWYPRZYCISK)	—	MOUSEMIDDLEBUTTON (MMB)
ŚRODKOWYPRZYCISK (ŚPM)	—	MOUSEMIDDLEBUTTON (MMB)
TASUJ	—	SHUFFLE
TEKSTY (TS)	—	TEXTSCREEN (TS)
TG	—	TAN
TŁO	—	BACKGROUND (BG)
TŁOTEKSTU (TT)	—	TEXTBACKGROUND (TG)
TREŚĆ	—	TEXT
TRYBDRUKU	—	SETCOPYMODE (SCM)
TRYBWZORU	—	FILLTYPE
TS (TEKSTY)	—	TEXTSCREEN (TS)
TT (TŁOTEKSTU)	—	TEXTBACKGROUND (TG)
US (USUŃ)	—	ERASE (ER)
USUŃ (US)	—	ERASE (ER)
USUŃPLIK	—	ERASEFILE (ERF)
USW	—	ERALL
USWP	—	ERP
USWZ	—	ERNS
USZ	—	ERN
WAR (WARTOŚĆ)	—	THING
WARTOŚĆ (WAR)	—	THING
WL	—	UPPERCASE (UC)
WŁĄCZDRUK	—	PRINTON (PON)
WŁĄCZŚLAD	—	TRACE

WOLNE	—	.FREESPACE
WPISZ	—	TYPE
WRÓĆ	—	HOME
WS (WSTECZ)	—	BACK (BK)
WSTECZ (WS)	—	BACK (BK)
WY (WYNIK)	—	OUTPUT (OP)
WYŁĄCZDRUK	—	PRINTOFF (POF)
WYŁĄCZŚLAD	—	UNTRACE
WYNIK (WY)	—	OUTPUT (OP)
WZÓR	—	FILLPATTERN
WZÓR16	—	FILLPATT16
WZÓRKRESKI	—	LINEPATTERN
ZAMALUJ	—	FILL
ZAPISZ	—	SAVEALL
ZAPISZO	—	SAVESCR
ZAPISZOKNO	—	SAVEWND
ZAPISZP	—	SAVE
ZAPISZRED	—	SAVED
ZAPISZW	—	SAVEOBJ
ZD (ZDANIE)	—	SENTENCE (SE)
ZDANIE (ZD)	—	SENTENCE (SE)
ZMAŻ	—	CLEAN
ZMAŻOKNO (ZO)	—	CLEANWINDOW (CLW)
ZMAŻTEKST (ZT)	—	CLEARTEXT
ZMIENNA?	—	NAMEP
ZNAK	—	CHAR
ZO (ZMAŻOKNO)	—	CLEANWINDOW (CLW)
ZRÓB	—	RUN
ZT (ZMAŻTEKST)	—	CLEARTEXT
ŻÓŁW?	—	TURTLESTATE
.ELIPSA	—	.ELIPSE
.GENERATOR	—	.GENER
.KOŁO	—	.CIRCLE
.KWADRAT	—	.SQUARE
.ŁADUJBIN	—	.BLOAD
.ŁAMANA	—	.MULTILINE
.ŁUK	—	.BOW
.NUMERBŁĘDU	—	.ERRORNUMBER
.ODŚMIECANIE	—	.INSTAL

.OKNO?	—	.WINDOWSIZE (.WSIZE)
.PIERWOTNE	—	.PRIMITIVES
.POBŁĄD	—	.POERR
.POBŁĘDY	—	.POERRORS
.POBŁĘDZIE	—	.ONERROR
.PROSTOKĄT	—	.RECTANGLE
.RS232	—	.SERIALMODE
.RS232?	—	.SERIALACTIVE (.SA)
.SYSTEM	—	.SYSTEM
.SZYBKOŚĆ	—	.SPEED
.TRÓJKĄT	—	.TRIANGLE
.UMIEŚĆ	—	.DEPOSIT
.UMIEŚĆP	—	.OUT
.WIELOKĄT	—	.POLYGON
.WYCINEK	—	.PIESLICE
.WYWOŁAJ	—	.CALL
.ZAJĘTE	—	.RESERVED
.ZAJMIJ	—	.RESERVE
.ZAPISZBIN	—	.BSAVE
.ZOBACZ	—	.EXAMINE
.ZOBACZP	—	.IN
.ZWOLNIJ	—	.FREE

Dodatek D.
SŁOWNICZEK ANGIELSKO-POLSKI NAZW PROCEDUR PIERWOTNYCH

Dodatek ten zawiera spis angielskich nazw procedur pierwotnych języka AC-LOGO wraz z ich polskimi odpowiednikami.

ABS	—	ABS
AND	—	I
ARCCOS	—	ARCCOS
ARCSIN	—	ARCSIN
ARCTAN	—	ARCTG
ASCII	—	ASCII
BACK (BK)	—	WSTECZ (WS)
BACKGROUND (BG)	—	TŁO
BC (BORDERCOLOR)	—	RAMKA
BF (BUTFIRST)	—	BEZPIERW (BP)
BG (BACKGROUND)	—	TŁO
BK (BACK)	—	WSTECZ (WS)
BL (BUTLAST)	—	BEZOST (BO)
BOF (BRIGHTOFF)	—	NIEJASKRAWO
BON (BRIGHTON)	—	JASKRAWO
BORDERCOLOR (BC)	—	RAMKA
BRIGHTOFF (BOF)	—	NIEJASKRAWO
BRIGHTON (BON)	—	JASKRAWO
BUTFIRST (BF)	—	BEZPIERW (BP)
BUTLAST (BL)	—	BEZOST (BO)
BYE	—	DOŚĆ
CAT	—	KATALOG
CHAR	—	ZNAK
CLEAN	—	ZMAŻ
CLEANWINDOW (CLW)	—	ZMAŻOKNO (ZO)

CLEARSCREEN (CS)	—	CZYŚĆ (CS)
CLEARTEXT (CT)	—	ZMAŻTEKST (ZT)
CLW (CLEANWINDOW)	—	ZMAŻOKNO (ZO)
COPYSCREEN (CPS)	—	DRUKUJOBRAZ
COPYWINDOW (CPW)	—	DRUKUJOKNO
COS	—	COS
COUNT	—	DŁUGOŚĆ
CPS (COPYSCREEN)	—	DRUKUJOBRAZ
CPW (COPYWINDOW)	—	DRUKUJOKNO
CS (CLEARSCREEN)	—	CZYŚĆ (CS)
CT (CLEARTEXT)	—	ZMAŻTEKST (ZT)
CURSOR	—	KURSOR?
DATE	—	DATA
DEFINE	—	OKREŚL
DEFINEDP	—	OKREŚLONE?
DEFINEWINDOW (DFW)	—	OPISOKNA
DELAY	—	OPÓŹNIENIE
DFW (DEFINEWINDOW)	—	OPISOKNA
DOT	—	PKT
DRIVE	—	DYSK?
ED (EDIT)	—	RED
EDALL	—	REDW
EDIT (ED)	—	RED
ELSE	—	INACZEJ
EMPTYP	—	PUSTE?
END	—	JUŻ
EQUALP	—	RÓWNE?
ER (ERASE)	—	USUŃ (US)
ERALL	—	USW
ERASE (ER)	—	USUŃ (US)
ERASEFILE (ERF)	—	USUŃPLIK
ERF (ERASEFILE)	—	USUŃPLIK
ERN	—	USZ
ERNS	—	USWZ
ERPS	—	USWP
EXP	—	EXP
FALSE	—	FAŁSZ
FCAT	—	PLIKI
FD (FORWARD)	—	NAPRZÓD (NP)
FENCE	—	PŁOT
FILL	—	ZAMALUJ

FILLPATT16	— WZÓR16
FILLPATTERN	— WZÓR
FILLTYPE	— TRYBWZORU
FIRST	— PIERW
FLASHOFF (FOF)	— NIEMIGAJĄCO
FLASHON (FON)	— MIGAJĄCO
FOF (FLASHOFF)	— NIEMIGAJĄCO
FON (FLASHON)	— MIGAJĄCO
FORWARD (FD)	— NAPRZÓD (NP)
FPUT	— NAP
FS (FULLSCREEN)	— GRAFIKA (GR)
FULLSCREEN (FS)	— GRAFIKA (GR)
HEADING	— KĄT?
HIDEMOUSE (HM)	— SM
HIDETURTLE (HT)	— SŻ
HM (HIDEMOUSE)	— SM
HOME	— WRÓĆ
HT (HIDETURTLE)	— SŻ
IF	— JEŚLI
INT	— ENT
ITEM	— ELEMENT
KEYPRES (KP)	— KLAWISZ?
KP (KEYPRES)	— KLAWISZ?
LAST	— OST
LC (LOWERCASE)	— ML
LEFT (LT)	— LEWO (LW)
LINEPATTERN	— WZÓRKRESKI
LIST	— LISTA
LISTP	— LISTA?
LN	— LN
LOAD	— ŁADUJ
LOADED	— ŁADUJRED
LOADOBJ	— ŁADUJW
LOADSCR	— ŁADUJO
LOADWND	— ŁADUJOKNO
LOCAL	— LOKALNA
LOWERCASE (LC)	— ML
LPUT	— NAK
LT (LEFT)	— LEWO (LW)
MAKE	— PRZYPISZ (PRZYP)
MEMBERP	— ELEMENT?

MLB (MOUSELEFTBUTTON)	—	LEWYPRZYCISK (LPM)
MMB (MOUSENIDDLEBUTTON)	—	ŚRODKOWYPRZYCISK (ŚPM)
MOUSELEFTBUTTON (MLB)	—	LEWYPRZYCISK (LPM)
MOUSEMIDDLEBUTTON (MMB)	—	ŚRODKOWYPRZYCISK (ŚPM)
MOUSEPOSITION (MPOS)	—	POZMYSZKI?
MOUSERIGHTBUTTON (MRB)	—	PRAWYPRZYCISK (PPM)
MPOS (MPOSITION)	—	POZMYSZKI?
MRB (MOUSERIGHTBUTTON)	—	PRAWYPRZYCISK (PPM)
NAMEP	—	ZMIENNA?
NOT	—	NIE
NUMBERP	—	LICZBA?
OP (OUTPUT)	—	WYNIK (WY)
OR	—	LUB
OUTPUT (OP)	—	WYNIK (WY)
PATH	—	ŚCIEŻKA?
PC (PENCOLOR)	—	PISAK
PD (PENDOWN)	—	OPUŚĆ (OPU)
PE (PENERASE)	—	ŚCIERANIE (ŚCIER)
PENCOLOR (PC)	—	PISAK
PENDOWN (PD)	—	OPUŚĆ (OPU)
PENERASE (PE)	—	ŚCIERANIE (ŚCIER)
PENREVERSE (PX)	—	ODWRACANIE (ODWR)
PENUP (PU)	—	PODNIEŚ (POD)
PI	—	PI
PO (PRINTOUT)	—	PO
POALL	—	POW
POF (PRINTOFF)	—	WYŁĄCZDRUK
POINTCOLOR	—	KOLORP?
POINTP	—	PUNKT?
PON (PRINTON)	—	WŁĄCZDRUK
PONS	—	POWZ
POPS	—	POWP
POS (POSITION)	—	POZ?
POSITION (POS)	—	POZ?
POTS	—	PONP
PR (PRINT)	—	PISZ (PS)
PRIMITIVEP	—	PIERWOTNE?
PRINT (PR)	—	PISZ (PS)
PRINTOFF (POF)	—	WYŁĄCZDRUK
PRINTON (PON)	—	WŁĄCZDRUK
PRINTOUT (PO)	—	PO

PRODUCT	— ILOCZYN
PU (PENUP)	— PODNIEŚ (POD)
PX (PENREVERSE)	— ODWRACANIE (ODWR)
QUOTIENT	— ILORAZC
RANDOM	— LOSOWA
RC (READCHAR)	— CZYTAJZNAK (CZ)
READCHAR (RC)	— CZYTAJZNAK (CZ)
READLIST (RL)	— CZYTAJLISTĘ (CL)
READSERIALCHAR (RSC)	— CZYTAJRS232 (CRS)
RECYCLE	— ODŚMIEĆ
REMAINDER	— RESZTA
REPEAT	— POWTÓRZ
RERANDOM	— LOSUJ
RIGHT (RT)	— PRAWO (PW)
RL (READLIST)	— CZYTAJLISTĘ (CL)
RSC (READSERIALCHAR)	— CZYTAJRS232 (CRS)
RT (RIGHT)	— PRAWO (PW)
RUN	— ZRÓB
SAVE	— ZAPISZP
SAVEALL	— ZAPISZ
SAVED	— ZAPISZRED
SAVEOBJ	— ZAPISZW
SAVESCR	— ZAPISZO
SAVEWND	— ZAPISZOKNO
SCM (SETCOPYMODE)	— TRYBDRUKU
SCO(SETCOPYORIENTATION)	— ORIENTUJDRUK
SCRUNCH	— PROPORCJA?
SCS (SETCOPYSCALE)	— SKALUJDRUK
SE (SENTENCE)	— ZDANIE (ZD)
SENTENCE (SE)	— ZDANIE (ZD)
SERIALOUT	— PISZRS232
SETCOPYMARGIN	— MARGINESYDRUKU
SETCOPYMODE (SCM)	— TRYBDRUKU
SETCOPYORIENTATION(SCO)	— ORIENTUJDRUK
SETCOPYSCALE (SCS)	— SKALUJDRUK
SETCURSOR	— KURSOR
SETDRIVE	— DYSK
SETH (SETHEADING)	— KĄT
SETHEADING (SETH)	— KĄT
SETPATH	— ŚCIEŻKA
SETSCRUNCH	— PROPORCJA

SETSPLIT	—	DZIELEKRAN
SETX	—	POZX
SETXY	—	POZ
SETY	—	POZY
SHOW	—	POKAŻ
SHOWMOUSE (SM)	—	PM
SHOWTURTLE (ST)	—	PŻ
SHUFFLE	—	TASUJ
SIGN	—	SIGN
SIN	—	SIN
SM (SHOWMOUSE)	—	PM
SORT	—	SORTUJ
SOUND	—	NUTA
SQRT	—	PIERWIASTEK (PWK)
ST (SHOWTURTLE)	—	PŻ
STARTTIMER	—	ODLICZANIE
STOP	—	STOP
SUM	—	SUMA
TAN	—	TG
TC (TEXTCOLOR)	—	KOLORTEKSTU (KT)
TEXT	—	TREŚĆ
TEXTBACKGROUND (TG)	—	TŁOTEKSTU (TT)
TEXTCOLOR (TC)	—	KOLORTEKSTU (KT)
TEXTSCREEN (TS)	—	TEKSTY (TS)
TG (TEXTBACKGROUND)	—	TŁOTEKSTU (TT)
THING	—	WARTOŚĆ (WAR)
TIME	—	CZAS
TO	—	OTO
TOPLEVEL	—	PRZERWIJ
TOWARDS	—	AZYMUT
TRACE	—	WŁĄCZŚLAD
TRUE	—	PRAWDA
TS (TEXTSCREEN)	—	TEKSTY (TS)
TURTLESTATE	—	ŻÓŁW?
TYPE	—	WPISZ
UC (UPPERCASE)	—	WL
UNTRACE	—	WYŁĄCZŚLAD
UPPERCASE (UC)	—	WL
WAIT	—	CZEKAJ
WINDOW	—	OKNO
WORD	—	SŁOWO

WORDP	—	SŁOWO?
WRAP	—	SKLEJ
XCOR	—	POZX?
YCOR	—	POZY?
.BLOAD	—	.ŁADUJBIN
.BOW	—	.ŁUK
.BSAVE	—	.ZAPISZBIN
.CALL	—	.WYWOŁAJ
.CIRCLE	—	.KOŁO
.DEPOSIT	—	.UMIEŚĆ
.ELLIPSE	—	.ELIPSA
.ERRORNUMBER	—	.NUMERBŁĘDU
.EXAMINE	—	.ZOBACZ
.FREE	—	.ZWOLNIJ
.FREESPACE	—	.WOLNE
.GENER	—	.GENERATOR
.IN	—	.ZOBACZP
.INSTAL	—	.ODŚMIECANIE
.MULTILINE	—	.ŁAMANA
.ONERRROR	—	.POBŁĘDZIE
.OUT	—	.UMIEŚĆP
.PIESLICE	—	.WYCINEK
.POERR	—	.POBŁĄD
.POERRORS	—	.POBŁĘDY
.POLYGON	—	.WIELOKĄT
.PRIMITIVES	—	.PIERWOTNE
.RECTANGLE	—	.PROSTOKĄT
.RESERVE	—	.ZAJMIJ
.RESERVED	—	.ZAJĘTE
.SA (.SERIALACTIVE)	—	.RS232?
.SERIALACTIVE (.SA)	—	.RS232?
.SERIALMODE	—	.RS232
.SPEED	—	.SZYBKOŚĆ
.SQUARE	—	.KWADRAT
.SYSTEM	—	.SYSTEM
.TRIANGLE	—	.TRÓJKĄT
.WINDOWSIZE (.WSIZE)	—	.OKNO?
.WSIZE (.WINDOWSIZE)	—	.OKNO?

Dodatek E.
SPIS KOMUNIKATÓW BŁĘDÓW

W opisie komunikatów błędów użyto następujących oznaczeń:

P, R — nazwy procedur pierwotnych;
X, Y — nazwy procedur zadeklarowanych przez użytkownika;
O — operator;
Z — nazwa zmiennej;
D — dowolna dana;
N — numer poziomu wywołania procedury, na którym wystąpił błąd;
M — numer wiersza w tekście procedury, w którym wystąpił błąd.

BŁĘDY ŁADOWANIA LOGO

Brak minimalnej wolnej pamięci.
Program LOGO nie może być uruchomiony.

Komunikat występuje po wywołaniu LOGO, jeśli w pamięci operacyjnej komputera nie ma już miejsca na program LOGO i jego dane. Brak miejsca w pamięci może być wynikiem zbyt małej pamięci operacyjnej w komputerze (np. 256 kB zamiast pełnych 640 kB), lub częściej, załadowaniem do pamięci zbyt dużej liczby programów "rezydentnych" (na przykład SideKick, program obsługi myszki, programy śledzące — debuggery itp.). W celu usunięcia błędu należy usunąć zbędne programy rezydentne (por. [11]) i uruchomić program LOGO jeszcze raz.

Brak systemowego generatora znaków.
Program LOGO nie może być uruchomiony.

Brak podstawowego generatora znaków.
Program LOGO nie może być uruchomiony.

Komunikaty te występują podczas ładowania LOGO, gdy nie można odszukać na dysku plików SYSTEM.FNT i/lub LAT8X14.FNT z systemowym

i podstawowym generatorem znaków. Błąd ten może wystąpić w dwóch sytuacjach:

- gdy któryś z wymienionych plików został omyłkowo wymazany z dysku,
- zmieniono nazwę katalogu systemowego w pliku konfiguracyjnym i zapomniano o skopiowaniu do tego katalogu plików z generatorami znaków.

W celu usunięcia błędu należy skopiować pliki z generatorami znaków do katalogu systemowego (standardowo do katalogu, w którym znajduje się plik ACLOGO.EXE) i ponowić próbę uruchomienia systemu.

Nierozpoznana karta graficzna.
Program LOGO nie może być uruchomiony.

Komunikat występuje podczas próby wczytania LOGO na komputerze, który nie jest wyposażony w kartę graficzną Hercules HGC, EGA lub VGA (SVGA). Program LOGO nie może być uruchomiony na komputerze, który nie jest wyposażony w jedną z wymienionych kart graficznych.

BŁĘDY SPRZĘTOWE

Błędy te powstają na skutek niepoprawnej komunikacji z urządzeniami zewnętrznymi (pamięci dyskowe, drukarki itp.). Po wystąpieniu błędu, na ekranie jest wyświetlane specjalne okienko, w którym jest wydrukowany krótki opis błędu oraz komunikat:

Powtórz Zaniechaj Ignoruj

Wybranie (za pomocą klawiatury — naciśnięcie wyróżnionego klawisza, lub za pomocą myszki — przesunięcie kursora myszki nad wybraną opcję i naciśnięcie lewego klawisza) opcji "Powtórz" powoduje ponowienie próby wykonania operacji, która spowodowała błąd. Wybranie opcji "Zaniechaj" powoduje przerwanie wykonywania programu LOGO i wywołanie procedur obsługi błędu (standardowo wyświetlenie komunikatu na ekranie, por. procedura pierwotna .POBŁĘDZIE, ang. .ONERROR). Wybranie opcji "Ignoruj" powoduje zignorowanie błędu i powrót do wykonywania programu LOGO.

BŁĘDY WYKONYWANIA PROGRAMU

Standardowe sposoby obsługi niżej opisanych błędów wykonywania programu można zmieniać za pomocą procedur .POBŁĄD, .POBŁĘDY i .POBŁĘDZIE (ang. .POERR, .POERRORS i .ONERROR), opisanych w rozdziale 8.

Błąd 1 — poprawne zakończenie pracy programu LOGO

Błąd numer 1 powoduje zakończenie pracy programu (ciągu procedur) LOGO i powrót do edytora wierszowego bez drukowania komunikatu.

Błąd 2 — zakończenie pracy programu LOGO i wyjście do systemu MS-DOS

Błąd numer 2 jest generowany po wykonaniu procedur pierwotnych DOŚĆ lub .SYSTEM (ang. BYE i .SYSTEM). Błąd ten powoduje zakończenie wykonywania programu LOGO i powrót na poziom systemu operacyjnego MS-DOS połączony z utratą wszystkich procedur i zmiennych LOGO.

Błąd 3 — przerwanie wykonywania programu LOGO

Błąd numer 3 występuje po wykonaniu procedury pierwotnej PRZERWIJ (ang. TOPLEVEL), po wydaniu w edytorze ekranowym polecenia Ctrl-KQ (zaniechania edycji) lub po wciśnięciu klawiszy Ctrl i Break. Standardowo błąd ten powoduje wyświetlenie komunikatu:

*** Przerwanie ***

*** Przerwanie na poziomie N w wierszu M w procedurze Y.

Błąd 4 — brak wolnej pamięci operacyjnej dla nowego obiektu

Błąd numer 4 występuje w sytuacji, gdy (pomimo uporządkowania pamięci) nie ma wolnego miejsca na utworzenie nowego obiektu. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Brak wolnej pamięci.

Brak wolnej pamięci na poziomie N w wierszu N w procedurze Y.

Błąd 5 — brak definicji procedury

Błąd numer 5 występuje w sytuacji, gdy nastąpiła próba wykonania niezdefiniowanej procedury, lub nazwy procedury nie oddzielono separatorem od parametru. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Nie wiem jak zrobić X.

Nie wiem jak zrobić X na poziomie N w wierszu M w procedurze Y.

Błąd 6 — nieoczekiwany koniec tekstu procedury

Błąd numer 6 występuje w sytuacji, gdy niepoprawnie zdefiniowano procedurę przy pomocy OKREŚL (ang. DEFINE). Najczęstszą przyczyną tego błędu jest pominięcie słowa JUŻ (ang. END) na końcu definicji procedury. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Nieoczekiwany koniec tekstu procedury.

Nieoczekiwany koniec tekstu procedury na poziomie M w wierszu N w procedurze Y.

Błąd 7 — próba wykonania operacji arytmetycznej poza wyrażeniem

Błąd numer 7 występuje w sytuacji, gdy którykolwiek z operatorów (np.+, −, * itp.) został użyty w procedurze poza kontekstem analizy wyrażenia. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Próba użycia O poza wyrażeniem.

Próba użycia O poza wyrażeniem na poziomie N w wierszu M w procedurze Y.

Błąd 8 — brak analizy wyniku procedury/funkcji

Błąd ten występuje w sytuacji, gdy parametr wynikowy procedury użytkownika (uzyskany za pomocą procedury WYNIK — ang. OUTPUT) nie został pobrany do dalszej analizy. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Nie powiedziałeś co zrobić z D.

Nie powiedziałeś co zrobić z D na poziomie N w wierszu M w procedurze Y.

Błąd 9 — brak parametru dla procedury

Błąd ten występuje w sytuacji, gdy liczba parametrów aktualnych procedury jest mniejsza niż liczba jej parametrów formalnych. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Brak parametru dla D.

Brak parametru dla D na poziomie N w wierszu M w procedurze Y.

Błąd 10 — błędny parametr procedury lub funkcji pierwotnej

Błąd ten występuje w sytuacji, gdy typ parametru procedury lub funkcji pierwotnej nie jest zgodny z przewidywanym (na przykład parametr procedury SŁOWO — ang. WORD — jest listą). Standardowo błąd ten powoduje wyświetlenie komunikatu:

Błędny parametr P.

Błędny parametr P na poziomie N w wierszu M w procedurze Y.

Błąd 11 — procedura pierwotna nie jest funkcją

Błąd ten występuje w sytuacji, gdy w wyrażeniu użyto procedurę pierwotną nie będącą funkcją, lub po wywołaniu procedury R z niewystarczającą liczbą parametrów, po którym nastąpiło wywołanie procedury P. Standardowo błąd ten powoduje wyświetlenie komunikatu:

P nie daje wyniku potrzebnego R.

P nie daje wyniku potrzebnego R na poziomie N w wierszu M w procedurze Y.

Błąd 12 — zmienna nie ma zdefiniowanej wartości

Błąd ten występuje w sytuacji, gdy odwołano się do nieistniejącej

zmiennej lub do zmiennej lokalnej o niezdefiniowanej wartości. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Z nie ma zdefiniowanej wartości.

Z nie ma zdefiniowanej wartości na poziomie N w wierszu M w procedurze Y.

Błąd 13 — niezgodność nawiasów

Błąd ten występuje w sytuacji, gdy w danym wyrażeniu liczba nawiasów lewostronnych jest większa niż prawostronnych. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Niezgodność nawiasów.

Niezgodność nawiasów na poziomie N w wierszu M w procedurze Y.

Błąd 14 — procedura pierwotna nie jest zachłanna

Błąd ten występuje przy próbie umieszczenia w nawiasach wywołania procedury P nie będącej procedurą zachłanną. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Procedura pierwotna P nie jest procedurą zachłanną.

Procedura pierwotna P nie jest procedurą zachłanną na poziomie N w wierszu M w procedurze Y.

Błąd 15 — zbyt złożone wyrażenie

Błąd ten występuje przy próbie wywołania procedury zachłannej z większą liczbą parametrów niż 127. Może on także wystąpić podczas analizy bardzo skomplikowanych wyrażeń arytmetycznych, mających więcej niż 127 operacji o różnych priorytetach. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Zbyt złożone wyrażenie.

Zbyt złożone wyrażenie na poziomie N w wierszu M w procedurze Y.

Błąd 16 — dzielenie przez zero

Błąd występuje po próbie dzielenia przez zero. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Próba dzielenia przez zero.

Próba dzielenia przez zero na poziomie N w wierszu M w procedurze Y.

Błąd 17 — użycie OTO poza edytorem wierszowym

Błąd ten występuje po wywołaniu procedury pierwotnej OTO (ang. TO) w innym kontekście niż na początku wiersza redagowanego za pomocą edytora wierszowego. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Niedozwolone użycie TO.

Niedozwolone użycie TO na poziomie N w wierszu M w procedurze Y.

Błąd 18 — próba dostępu do niezarezerwowanego obszaru

Błąd ten następuje po odwołaniu, za pomocą procedur .UMIEŚĆ, .ZOBACZ i .WYWOŁAJ (ang. .DEPOSIT, .EXAMINE i .CALL), do obszaru niezarezerwowanego uprzednio za pomocą procedury .ZAJMIJ (ang. RESERVE). Standardowo błąd ten powoduje wyświetlenie komunikatu:

Próba dostępu do niezarezerwowanego obszaru.

Próba dostępu do niezarezerwowanego obszaru na poziomie N w wierszu M w procedurze Y.

Błąd 19 — brak wskazanego pliku

Błąd ten występuje po próbie odczytu nieistniejącego pliku. Błąd ten może być spowodowany albo podaniem błędnej nazwy pliku, albo błędnej nazwy dysku lub katalogu. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Brak pliku.

Brak pliku na poziomie N w wierszu M w procedurze Y.

Błąd 20 — błędny format pliku graficznego

Błąd ten występuje podczas próby ładowania obrazu ekranu z pliku o nieznanym formacie, nie odpowiadającym formatowi AC-LOGO. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Błędny format pliku.

Błędny format pliku na poziomie N w wierszu M w procedurze Y.

Błąd 21 — zbyt długi plik

Błąd ten występuje podczas próby załadowania pliku, który nie mieści się w całości w obszarze roboczym edytora ekranowego (standardowo obszar ten obejmuje 64 kB pamięci). W celu wyeliminowania tego błędu należy usunąć wszelkie zbędne procedury i funkcje, usunąć tekst z obszaru roboczego edytora i następnie ponowić próbę wczytania pliku. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Plik zbyt długi.

Plik zbyt długi na poziomie N w wierszu M w procedurze Y.

Błąd 22 — błąd utworzenia/zapisu pliku

Błąd ten powstaje podczas zapisu pliku na całkowicie zapełniony dysk lub na dysk z włączoną blokadą zapisu (por. [11]). Po wystąpieniu błędu można odblokować blokadę zapisu dysku lub usunąć zbędne pliki (na przykład za pomocą procedury USUŃPLIK, ang. ERASEFILE). Standardowo błąd ten powoduje wyświetlenie komunikatu:

Nie można zapisać pliku.

Nie można zapisać pliku na poziomie N w wierszu M w procedurze Y.

Błąd 23 — żółw poza oknem graficznym

Błąd ten powstaje w wyniku próby przemieszczenia żółwia poza okno graficzne, jeśli okno znajduje się w trybie "płot". Standardowo błąd ten powoduje wyświetlenie komunikatu:

Żółw poza oknem.

Żółw poza oknem na poziomie N w wierszu M w procedurze Y.

Błąd 24 — błąd transmisji przez złącze szeregowe RS232C

Błąd ten powstaje w wyniku próby transmisji przez złącze szeregowe, gdy do złącza nie jest dołączony inny komputer. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Błędne połączenie przez złącze szeregowe RS-232C.

Błędne połączenie przez złącze szeregowe RS-232C na poziomie N w wierszu M w procedurze Y.

Błąd 25 — błąd transmisji znaku na drukarkę

Błąd ten powstaje w wyniku próby wydrukowania kopii ekranu lub tekstu, gdy drukarka nie jest dołączona, nie jest włączona lub nie ma zapalonej lampki "READY" (wskaźnika gotowości do wydruku). W celu usunięcia błędu należy dołączyć drukarkę, włączyć ją i ewentualnie nacisnąć przycisk "READY" zapalając lampkę gotowości drukarki. Standardowo błąd ten powoduje wyświetlenie komunikatu:

Błąd wysyłania znaku na drukarkę.

Błąd wysyłania znaku na drukarkę na poziomie N w wierszu M w procedurze Y.

BŁĘDY KOMPILACJI

Brak słowa OTO rozpoczynającego procedurę

Komunikat występuje po próbie kompilacji procedury bez nagłówka, lub procedury z wielokrotnie użytym słowem JUŻ.

Brak nazwy procedury po OTO

Komunikat występuje po próbie kompilacji procedury z pustą nazwą lub o nazwie będącej operatorem.

Nie można redefiniować procedury pierwotnej

Komunikat występuje po próbie kompilacji procedury o nazwie będącej nazwą procedury pierwotnej lub po opuszczeniu nazwy w nagłówku procedury bezparametrowej. W tym drugim przypadku, pierwsze słowo treści procedury jest traktowane jako jej nazwa.

Błędny parametr definiowanej procedury

Komunikat występuje po próbie kompilacji procedury, której parametr jest parametrem pustym (na przykład po znaku dwukropka początkującego parametr jest spacja).

INNE BŁĘDY

Brak wolnej pamięci dla utworzenia okienka

Komunikat występuje przy braku wolnej pamięci, potrzebnej do zapamiętania zawartości ekranu spod otwieranego okienka. Błąd braku pamięci jest najczęściej skutkiem udostępnienia zbyt małej pamięci dla LOGO (na przykład załadowania zbyt dużej liczby programów rezydentnych w systemie MS-DOS).

Użycie INACZEJ bez poprzedzającego JEŚLI

Błąd występuje w przypadku próby wykonania procedury INACZEJ (ang. ELSE) bez poprzedzającego ją wykonania procedury JEŚLI (ang. IF).

Nie można wydrukować tekstu edytora, naciśnij ESC

Błąd ten występuje w sytuacji, gdy jest niemożliwe wydrukowanie tekstu z obszaru roboczego edytora, a użytkownik po stwierdzeniu błędu drukowania wybrał opcję "Zaniechaj". Naciśnięcie ESC powoduje powrót do systemu okienkowego.

Próba pobrania niezapisanego stosu rozbioru
Błąd w optymalizacji TRO
Nie można porównywać obiektów o różnych typach
Nieznany błąd, nie znam jego przyczyny???

Błędy te nie powinny wystąpić podczas normalnej pracy systemu LOGO. Ich wystąpienie świadczy najczęściej o błędnej pracy komputera, przekłamaniach pamięci, błędnym wczytaniu programu ACLOGO.EXE itp. W przypadku ich wystąpienia należy przede wszystkim sprawdzić poprawność działania sprzętu komputera, a w szczególności jego pamięci operacyjnej.

Dodatek F.
OPIS PLIKU KONFIGURACYJNEGO

W AC-LOGO wprowadzono możliwość konfigurowania systemu, czyli dostosowywania go do indywidualnych potrzeb każdego użytkownika. Służy do tego celu plik konfiguracyjny o nazwie LOGO.CFG, znajdujący się w katalogu, z którego jest wywoływany program ACLOGO.EXE. Jest to plik tekstowy. Każdy wiersz tego pliku zawiera jedno polecenie. Nazwy poleceń są zawsze dwuznakowe i zawsze muszą być umieszczone na początku wiersza. Wiersz, który nie zaczyna się dwuznakową nazwą polecenia jest traktowany jako komentarz i pomijany. Nazwa polecenia może być zapisana małymi lub wielkimi literami. Po nazwie polecenia znajdują się parametry. Parametrami mogą być liczby dziesiętne (ciągi cyfr od "0" do "9") i łańcuchy znaków (ciągi znaków od spacji do spacji lub końca wiersza). Liczba parametrów i ich typ jest zależna od polecenia. Pierwszy znak w wierszu, który nie może zostać rozpoznany jako poprawny parametr, jest traktowany jako początek komentarza — dalszy ciąg wiersza jest pomijany w analizie pliku.

Poniżej przedstawiono tabelę, w której opisano wszystkie polecenia pliku konfiguracyjnego uszeregowane alfabetycznie, z podaniem liczby i typu parametrów każdego polecenia.

Polecenie	Parametry		Wartość domyślna	Opis działania
	liczba	typ		
AL	1	słowo	AUTOEXEC.LOG	ustalenie nazwy pliku AUTOEXEC.LOG
AS	1	słowo	AUTOEXEC	ustalenie nazwy procedury startowej z pliku AUTOEXEC.LOG
DI	1	słowo	.\	ustalenie nazwy katalogu bieżącego
DP	1	liczba	15	ustalenie liczby wyświetlanych po przecinku cyfr (parametr z zakresu od 1 do 16)
DR	1	słowo	LDRAFT.PFT	ustalenie nazwy pliku pamiętającego definicje znaków narodowych dla drukarki pracującej w trybie druku niskiej jakości

Polecenie	Parametry		Wartość domyślna	Opis działania
	liczba	typ		
DS	1	słowo	. \	ustalenie nazwy katalogu systemowego, w którym powinny się znajdować pliki z generatorami znaków oraz pliki z definicjami znaków narodowych dla drukarki
DT	1	słowo	. \	ustalenie nazwy katalogu systemowego, w którym będą tworzone pliki tymczasowe systemu LOGO; jeśli system jest uruchamiany w sieci komputerowej, katalogi te MUSZĄ BYĆ RÓŻNE DLA KAŻDEGO UŻYTKOWNIKA
EA	1	liczba	1	ustalenie autotranslacji po ESC w edytorze: 0 — z pytaniem, 1 — bez pytania
EI	1	liczba	1	ustalenie trybu wpisywania znaków w edytorze: 0 — "zastępowanie", 1 — "wstawianie"
EN	1	liczba	1	ustalenie trybu wpisywania znaków w edytorze: 0 — "brak wcięć", 1 — "automatyczne stosowanie wcięć w tekście redagowanych procedur"
ET	1	liczba	8	ustalenie rozmiar tabulacji; po wciśnięciu klawisza TAB kursor jest przesuwany do co N-ej kolumny tekstu
G1	n	liczby	27 64	ustalenie sekwencji sterującej zerującej drukarkę
G2	n	liczby	27 65 8	ustalenie sekwencji sterującej wysyłanej przed włączeniem trybu graficznego
G3	n	liczby	27 65 12	ustalenie sekwencji sterującej wysyłanej po wyłączeniu trybu graficznego
G4	n	liczby	27 74 1	ustalenie sekwencji sterującej wysyłanej przed włączeniem trybu graficznego w trybie wysokiej jakości NLQ
G5	n	liczby	27 106 1	ustalenie sekwencji sterującej wysyłanej po wyłączeniu trybu graficznego w trybie wysokiej jakości NLQ
GE	1	słowo		wczytanie generatora znaków z pliku i nadanie mu kolejnego numeru
KB	15	liczby		zredefiniowanie jednego klawisza klawiatury (o numerze będącym pierwszym parametrem, odpowiadającym numerowi kla-

Polecenie	Parametry		Wartość domyślna	Opis działania
	liczba	typ		
				wisza w systemie operacyjnym MS-DOS). Następne parametry odpowiadają kodom znaków generatora z klawiszami sterującymi: — bez klawiszy sterujących, — z aktywnym klawiszem CapsLock, — z klawiszem SHIFT, — z SHIFT i CapsLock, — z ALT, — z ALT i CapsLock, — z CTRL, — z SHIFT i ALT, — z SHIFT, ALT i CapsLock, — z SHIFT, ALT i CTRL, — z SHIFT, ALT, CTRL i CLock, — z ALT i CTRL, — z ALT, CTRL i CLock, — aktywnym NumLock.
MU	1	liczba	1	ustal. trybu wyświetlania kursora myszki podczas wykonywania procedur LOGO: 1 — kursor niewidoczny na ekranie, 0 — kursor myszki widoczny.
NL	1	słowo	LNLQ.PFT	ustalenie nazwy pliku pamiętającego definicje znaków narodowych dla drukarki pracującej w trybie druku wysokiej jakości
P1	1	znak	63	ustalenie znaku zachęty dla edytora wierszowego
P2	1	znak	62	ustalenie znaku zachęty dla edytora wierszowego przy wczytywaniu kolejnych wierszy procedury definiowanej procedurą OTO (ang. TO)
P3	1	znak	32	ustalenie znaku zachęty dla edytora wierszowego przy wczytywaniu wiersza za pomocą CZYTAJLISTĘ (ang. READLINE)
PB	1	liczba	1	ustalenie trybu wydruku tekstu: 0 — "bez przerwy po stronie", 1 — "z przerwą po każdej stronie"
PC	n	liczby	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	ustalenie sposobu przekodowywania kolorów dla karty monochromatycznej i drukarki; wszystkie wymienione po nazwie polecenia kolory będą "widoczne" na ekranie HGC i na kopii ekranu na drukarce, natomiast wszystkie pozostałe nie będą widziane i drukowane

Polecenie	Parametry		Wartość domyślna	Opis działania
	liczba	typ		
PE	2	liczby	1 1	ustalenie skali (powiększenia) drukowania kopii ekranu na drukarce w kierunku poziomym i pionowym
PF	1	słowo		ustalenie tekstu drukow. w stopce każdej strony wysyłanej na drukarkę opcją "Drukuj tekst edytora". W tekście stopki można używać następujących znaków specjalnych: ^znak — wstawienie do tekstu stopki znaku sterującego o kodzie (znak — +40) \d — wstawienie do tekstu stopki aktualnej daty \p — wstawienie numeru aktualnie drukowanej strony \s — wstawienie znaku spacji \t — wstawienie aktualnego czasu
PG	2	liczby	70 50	ustalenie rozmiarów strony dla drukarki w znakach/wiersz i wierszach/stronę
PH	1	słowo		ustalenie tekstu drukowanego w nagłówku każdej strony wysyłanej na drukarkę opcją "Drukuj tekst edytora". W tekście nagłówka można używać wymienionych dla polecenia PF znaków specjalnych.
PM	2	liczby	1 1	ustalenie marginesów dla drukowania w spacjach od lewego brzegu i pustych wierszach od góry kartki
PN	1	liczba	0	ustalenie trybu drukowania: 0 — druk niskiej jakości DRAFT, 1 — druk wysokiej jakości NLQ.
PO	1	liczba	0	ustalenie orientacji wydruku kopii ekranu na drukarce: 0 — orient. normalna (PORTRAIT), 1 — orient. odwrócona (LANDSCAPE).
PS	1	słowo	LPT1	ustalenie nazwy drukarki w systemie operacyjnym MS-DOS
PT	1	liczba	6	ustalenie liczby druk. punktów/cal dla kopii ekranu na drukarce (numeru trybu graficznego drukarki) Parametr może przybierać wartości: — dla drukarek 9-igłowych: 0 — rozdzielczość 60 punktów/cal 1 — rozdzielczość 120 punktów/cal 2 — rozdzielczość 120 punktów/cal

Polecenie	Parametry		Wartość domyślna	Opis działania
	liczba	typ		3 — rozdzielczość 240 punktów/cal 4 — rozdzielczość 80 punktów/cal 5 — rozdzielczość 72 punktów/cal 6 — rozdzielczość 90 punktów/cal 7 — rozdzielczość 140 punktów/cal — dla drukarek 24-igłowych: 30 — rozdzielcz. 120 punktów/cal 31 — rozdzielcz. 240 punktów/cal 32 — rozdzielcz. 120 punktów/cal 33 — rozdzielcz. 240 punktów/cal 34 — rozdzielcz. 120 punktów/cal — dla drukarek laserowych: 75 — rozdzielcz. 75 punktów/cal 100 — rozdzielcz. 100 punktów/cal 150 — rozdzielcz. 150 punktów/cal 300 — rozdzielcz. 300 punktów/cal
RE	3	liczby	65535 10 1	ustalenie parametrów dla procedury porządkowania pamięci (por. opis proc. .ODŚMIECANIE — ang. .INSTALL)
SC	2	liczby	1 1	ustalenie proporcji ekranu w kierunku poziomym i pionowym. Polecenie to jest powiązane z procedurą pierwotną PROPORCJA/SETSCRUNCH — parametry procedury PROPORCJA są mnożone przez parametry polecenia SC. W ten sposób można "zmienić" rozmiary ekranu, dopasowując je na przykład do rozmiaru ekranu mikrokomputera ELWRO 800 Junior.
SE	5	liczby		ustalenie parametrów dla złącza transmisji szeregowej RS–232C: — numer złącza w MS-DOS + 1, — szybkość transmisji w bitach/s, — liczba bitów stopu, — liczba bitów znaków, — sposób kontroli błędów parzystości. Rolę poszczególnych parametrów wyjaśniono dokładniej w opisie procedury pierwotnej .RS232 (ang. .SERIALMODE).
SL	1	liczba	4	ustalenie liczby wierszy tekstu po wykonaniu procedury pierwotnej CS (por. opis procedury DZIELEKRAN, ang. SETSPLIT).
SP	1	liczba	0	ustalenie opóźnienia po wykonaniu każdej procedury pierwotnej (por. opis .SZYBKOŚĆ, ang. .SPEED).

Polecenie	Parametry		Wartość domyślna	Opis działania
	liczba	typ		
SS	1	słowo	COM2	ustalenie nazwy złącza transmisji szeregowej w syst. operacyjnym MS-DOS
T1	n	liczby	27 120 49	ustalenie sekwencji sterującej wysyłanej na drukarkę w celu włączenia druku wysokiej jakości NLQ
T2	n	liczby	27 120 48	ustalenie sekwencji sterującej wysyłanej na drukarkę w celu włączenia druku niskiej jakości DRAFT
TU	1	liczba	1	ustal. trybu wyświetlania żółwia: 0 — żółw niewidoczny na ekranie, 1 — żółw widoczny.
WC	6	liczby		ustal. dla okien błędów krytycznych: — koloru znaków w oknie, — koloru tła okna, — koloru znaków w ramce okna, — koloru tła w ramce okna, — koloru znaków w zaznacz. bloku, — koloru tła w zaznacz. bloku.
WE	7	liczby		ustalenie dla okna edytora: — koloru znaków w oknie, — koloru tła okna, — koloru znaków w ramce okna, — koloru tła w ramce okna, — koloru znaków w zaznacz. bloku, — koloru tła w zaznacz. bloku, — numeru generatora znaków.
WF	6	liczby		ustalenie dla okien, pojawiających się po wywołaniu pod-menu w systemie okienkowym: — koloru znaków w oknie, — koloru tła okna, — koloru znaków w ramce okna, — koloru tła w ramce okna, — koloru znaków w zaznacz. bloku, — koloru tła w zaznacz. bloku.
WR	4	liczby		ustalenie pozycji i rozmiarów okna edytora: — pozycja (x y) lewego, dolnego rogu okna, — szerokość okna, — wysokość okna w znakach.
WS	3	liczby		ustalenie pozycji i rozmiarów okna śledzenia: — szerokość/wysokość okna (w %), — podział okna w procentach,

Polecenie	Parametry		Wartość domyślna	Opis działania
	liczba	typ		
				— orientacja okna (0 — z lewej strony, 1 — u dołu ekranu).
WT	7	liczby		ustalenie dla okna śledzenia: — koloru znaków w oknie, — koloru tła okna, — koloru znaków w ramce okna, — koloru tła w ramce okna, — koloru znaków w zaznacz. bloku, — koloru tła w zaznacz. bloku, — numeru generatora znaków.
WU	7	liczby		ustalenie dla okna tekst.-graf.: — koloru znaków w oknie, — koloru tła okna, — koloru znaków w ramce okna, — koloru tła w ramce okna, — koloru znaków w zaznacz. bloku, — koloru tła w zaznacz. bloku, — numeru generatora znaków.
WW	6	liczby		ustalenie dla okien, pojawiających się po wywołaniu głównego menu w systemie okienkowym: — koloru znaków w oknie, — koloru tła okna, — koloru znaków w ramce okna, — koloru tła w ramce okna, — koloru znaków w zaznacz. bloku, — koloru tła w zaznacz. bloku,
XB	1	słowo	BAK	ustalenie standardowego rozszerzenia nazw plików — poprzednich kopii
XF	1	słowo	FNT	ustalenie standardowego rozszerzenia nazw plików z generat. znaków
XL	1	słowo	LOG	ustalenie standardowego rozszerzenia nazw plików z programami LOGO
XO	1	słowo	OBJ	ustalenie standardowego rozszerzenia nazw plików binarnych
XS	1	słowo	SCR	ustalenie standardowego rozszerzenia nazw plików pamiętających zawartość okienka (rysunki)
XT	1	słowo	TXT	ustalenie standardowego rozszerzenia nazw plików pamiętających zmienne LOGO
ZO	1	liczba	0	ustalenie opcji "Lupa": 0 — opcja "Lupa" nieaktywna, 1 — opcja "Lupa" włączona.

Na zakończenie podamy przykład pliku konfiguracyjnego z ustawionymi standardowymi wartościami parametrów.

```
;
;  plik konfiguracyjny LOGO
;  (c) AdvaCom 1992
;
al AUTOEXEC.LOG      ;  nazwa pliku AUTOEXEC
as AUTOEXEC          ;  nazwa procedury startowej AUTOEXEC
di C:\ACLOGO\POL     ;  katalog z plikami źródłowymi
dp 15                ;  liczba cyfr po przecinku w wydruku liczby
dr LDRAFT.PFT        ;  nazwa pliku ze znakami narodowymi dla DRAFT
ds C:\ACLOGO\        ;  katalog z plikami systemowymi
dt C:\ACLOGO\TMP\    ;  katalog na pliki tymczasowe
ea 1                 ;  automatyczna translacja po ESC w edytorze
ei 1                 ;  tryb edytora "wstawianie"
en 1                 ;  tryb edytora "automat. wcięcia tekstu"
et 8                 ;  rozmiar tabulacji
g1 27 64             ;  ESC '@' — zerowanie drukarki dla EPSON FX
g2 27 65 8           ;  ESC 'A' 8 — wejście do grafiki dla EPSON FX
g3 27 65 12          ;  ESC 'A' 12 — wyjście z grafiki dla EPSON FX
g4 27 74 1           ;  ESC J 1 — obniżenie głowicy o 1/216 cala
g5 27 106 1          ;  ESC j 1 — podwyższenie głowicy o 1/216 cala
ge ITL8X14           ;  wczytanie pliku z generatorem znaków numer 2
;kb N C1 ... C14     ;  bez definiowania wierszy opisu klawiatury
mu 1                 ;  kursor myszki widoczny
n1 LDRAFT.PFT        ;  nazwa pliku ze znakami narodowymi dla NLQ
p1 63                ;  podstawowy znak zachęty edytora — ?
p2 62                ;  znak zachęty dla definicji TO/END — >
p3 32                ;  znak zachęty dla CZYTAJLISTĘ/READLINE
                        — spacja
pb 0                 ;  drukowanie tekstu bez przerwy po stronie
pc 0                 ;  numery kolorów "widocznych" na
                        drukarce/HGC
pe 1 1               ;  powiększenie drukowania kopii ekranu
pf ^[W1\s\s\s\s\s\s\s\s\s\s\s\s\s\s\-\sStrona\s\p\s-^[W0    ;stopka
pg 70 50             ;  wymiary strony dla wydruku tekstu
ph ^[-1^[S0AC-LOGO\sAdvaCom\s\d\\ss\t^[T^[-0               ;nagłówek
pm 1 1               ;  marginesy dla drukowania
pn 0                 ;  włączenie druku niskiej jakości DRAFT
po 0                 ;  orientacja wydruku "normalna" (PORTRAIT)
```

```
ps LPT1                 ; nazwa drukarki w systemie MS-DOS
pt 1                    ; numer trybu graficznego dla drukarki EPSON
re 65535 10 1           ; parametry procedury .ODŚMIECANIE/.IN-
                          STALL
sc 1 1                  ; standardowe proporcje ekranu
;sc 1 0.75              ; proporcje ekranu dla karty EGA/VGA
;sc 1 0.64              ; proporcje ekranu dla karty Hercules
;sc 2.0 1.5             ; proporcje ekranu jak dla Junior-LOGO
;se 2 9600 2 8 0        ; złącze RS-232C
sl 4                    ; podział ekranu po wykonaniu CS
sp 0                    ; brak opóźnienia wykonywania procedur
ss COM2                 ; nazwa złącza transmisji szeregowej
                          w MS-DOS
t1 27 120 49            ; ESC x 1 — sekwencja wejścia w NLQ dla EPSON
t2 27 120 48            ; ESC x 0 — sekwencja wyjścia z NLQ dla EPSON
tu 1                    ; 1-żółw widoczny na ekranie
wc 0 15 0 15 15 0       ; kolory okien błędów krytycznych
we 15 0 15 0 15 0 1;    kolory okna edytora
wf 0 15 0 15 15 0       ; kolory okien pod-menu systemu okienkowego
wr 25 50 0              ; rozmiar i położenie okna śledzenia
ws 1 1 78 23            ; rozmiar okna edytora (cały ekran)
wt 15 0 15 0 15 0 1;    kolory okna śledzenia
wu 15 0 15 0 15 0 1;    kolory okna tekstowo-graficznego
ww 0 15 0 15 15 0       ; kolory okien menu głównego systemu okien-
                          kowego
xb BAK                  ; rozszerzenie nazwy dla plików *.BAK
xf FNT                  ; rozszerzenie nazwy dla plików z gen. znaków
xl LOG                  ; rozszerzenie nazwy dla plików z progr. LOGO
xo OBJ                  ; rozszerzenie nazwy dla plików binarnych
xs .SCR                 ; rozszerzenie nazwy dla plików z rysunkami
xt .TXT                 ; rozszerzenie nazwy dla plików ze zmiennymi
zo 0                    ; opcja "Lupa" nieaktywna
```

Dodatek G.
OPIS FORMATÓW PLIKÓW AC-LOGO

Opis pliku z generatorami znaków.

Każdy plik z generatorem znaków zawiera jeden generator, obejmujący od jednego do 256 znaków o jednakowej wielkości. Plik składa się z trzech części:

- statusowej o długości 128 bajtów,
- tabeli przeadresowań kodów znaków o długości 256 bajtów,
- wzorów kolejnych definiowanych znaków.

Wielkość każdego znaku może się zmieniać od 8 x 8 do 64 x 64 punkty, przy czym wielkość znaku w poziomie musi być zawsze wielokrotnością liczby 8. Każdy definiowany znak składa się z punktów w kolorze tła i atramentu, przy czym wartość "1" we wzorze znaku oznacza punkt w kolorze atramentu, a wartość "0" — punkt w kolorze tła. Znaki o rosnących numerach (kodach) są definiowane kolejno.

Struktura części statusowej pliku jest następująca:

- bajt określający typ pliku, standardowo równy #55,
- bajt określający typ generatora znaków, standardowo równy #AA,
- bajt określający szerokość każdego znaku w bajtach,
- bajt określający wysokość każdego znaku w liniach,
- dwa bajty określające długość części definiującej wzory znaków,
- nazwa generatora znaków (8 znaków),
- pozostałe bajty części statusowej nie są wykorzystane.

Druga część pliku zawiera tablicę przeadresowań kodów znaków. Jest ona adresowana kodami znaków i zawiera numery pozycji we wzorach znaków odpowiadające tym kodom. Na przykład, jeśli element tablicy przeadresowań o numerze #20 zawiera wartość 1, a wielkość znaku wynosi 16 x 8 punktów, to wybór znaku o kodzie #20 spowoduje narysowanie wzoru znaku pamiętanego jako drugi w trzeciej części pliku, począwszy od 17 bajtu tej

części. Wprowadzenie tablicy przekodowań umożliwia wprowadzenie niepełnych generatorów znaków, w których nie wszystkie wzory znaków są zdefiniowane. Pozwala to na dość znaczną oszczędność pamięci operacyjnej komputera.

Trzecia część pliku pamięta informacje o kolejnych zdefiniowanych wzorach znaków. Kolejne bajty wzoru każdego znaku są pamiętane wierszami znaku, na przykład:

bajt 1	*bajt 2*	*bajt M*
bajt M+1	*bajt M+2*	*bajt 2*M*
	. . .	
*bajt (N−1)*M+1*	*bajt (N−1)*M+2*	*bajt N*M*

Przykładowo, generator znaków 8 x 8 zawierający tylko dwa znaki: spację, drukowaną jako znak o kodzie 0 i znaki o kodach od 2 do 255 oraz znak strzałki, drukowany jako znak o kodzie 1, przedstawiono poniżej:

bajt typu pliku	#55
bajt typu generatora	#AA
szerokość znaku	1
wysokość znaku	8
długość generatora	16
nazwa generatora	PRZYKŁAD
dopełnienie zerami do 128 bajtów części statusowej	
pierwszy element tabeli przeadresowań	0
drugi element tabeli przeadresowań	1
pozostałe 254 elementy tabeli przeadresowań równe 0	
definicja spacji:	#00 #00 #00 #00 #00 #00 #00 #00
definicja strzałki:	#10 #30 #7f #ff #7f #30 #10 #00

Opis pliku z obrazem ekranu

Każdy plik z obrazem ekranu zawiera mapę bitową wybranego prostokątnego okna ekranu. Maksymalne rozmiary okna mogą obejmować cały ekran, natomiast minimalne — jeden bajt pamięci ekranu. Plik składa się z dwóch części:

- statusowej o rozmiarze 128 bajtów,
- mapy bitowej fragmentu ekranu, dla kart graficznych EGA/VGA podzielonej na poszczególne kolory; w tym ostatnim przypadku oddzielnie jest zapamiętywana mapa bitowa dla koloru czerwonego, zielonego i niebieskiego (ang. RED, GREEN i BLUE) oraz mapa bitowa rozjaśnienia kolorów (ang. BRIGHT).

Pliki graficzne pamiętające fragment ekranu dla karty graficznej EGA/VGA mogą być wczytywane na komputerze wyposażonym w kartę graficzną Hercules HGC. W tym wypadku mapy bitowe poszczególnych kolorów są sumowane. Podobnie, pliki zapisane na komputerze wyposażonym w kartę graficzną HGC są dla karty EGA/VGA wczytywane jednocześnie do wszystkich map bitowych poszczególnych kolorów (co daje na ekranie kolory biały i czarny).

Struktura części statusowej pliku jest następująca:

- bajt określający typ pliku, standardowo równy #33,
- bajt określający sposób pamiętania mapy bitowej ekranu:
  #F0 — mapa bitowa skompresowana,
  #E0 — mapa bitowa nieskompresowana;
- dwa bajty określające pozycję lewej krawędzi okna na ekranie,
- dwa bajty określające pozycję górnej krawędzi okna na ekranie,
- dwa bajty określające szerokość okna (w bajtach),
- dwa bajty określające wysokość okna (w liniach),
- dwa bajty określające liczbę map bitowych kolorów:
  #01 — jedna mapa bitowa, karta graficzna Hercules HGC,
  #04 — cztery mapy bitowe kolorów, karta graficzna EGA/VGA;
- cztery bajty zarezerwowane,
- do szesnastu znaków określających nazwę rysunku,
- pozostałe bajty części statusowej nie są wykorzystane.

Struktura pozostałej części pliku zależy od typu kompresji danych. Przy braku kompresji, w tej części pliku są pamiętane mapy bitowe kolejnych kolorów, bajt po bajcie. Standardowo wprowadzono kompresję danych. W takim przypadku druga część pliku zawiera ciąg bajtów określających kolejne mapy bitowe wg następującego algorytmu:

- ciąg do 127 bajtów równych pobranych z mapy bitowej ekranu jest zastępowany przez dwa bajty:
  — liczba bajtów równych,
  — powtarzający się bajt z pamięci ekranu;
- ciąg powyżej 127 bajtów równych z mapy bitowej ekranu jest zastępowany przez cztery bajty:
  — bajt #00,
  — dwa bajty określające liczbę powtarzających się bajtów z pamięci ekranu, pamiętane w konwencji "młodszy/starszy",
  — powtarzający się bajt z pamięci ekranu;
- ciąg do 127 bajtów różnych pobranych z mapy bitowej ekranu jest zastępowany przez ciąg:

— liczba bajtów różnych + #80,
— kolejne bajty;

- ciąg powyżej 127 bajtów różnych pobranych z mapy bitowej ekranu jest zastępowany przez ciąg:
  — bajt #80,
  — dwa bajty określające liczbę niepowtarzalnych bajtów z pamięci ekranu, pamiętane w konwencji "młodszy/starszy",
  — kolejne bajty.

Przyjęty sposób kompresji danych zapewnia dużą efektywność zapisu i odczytu plików pamiętających fragmenty ekranu, a także zmniejsza rozmiary plików (kilku-, a nawet kilkunastokrotnie).

Dodatek H.
ŚLEDZENIE RUCHU ŻÓŁWIA Z POZIOMU INNYCH PROGRAMÓW

W AC-LOGO wprowadzono możliwość "śledzenia" ruchów żółwia z poziomu innych programów rezydentnych, działających równolegle z systemem AC-LOGO. W tym celu udostępniono użytkownikowi możliwość korzystania z dwóch przerwań programowych mikroprocesora, których numery może on sam ustalać. Należy zauważyć, że dołączanie się do mechanizmu przerwań mikrokomputera IBM-PC jest bardzo niebezpieczne — niewłaściwe dołączenie powoduje najczęściej zawieszenie się komputera. Z tego względu lekturę tego rozdziału polecamy tylko bardzo zaawansowanym użytkownikom.

Pierwsze przerwanie jest uaktywniane w jednej z następujących sytuacji:

- przemieszczenie lub zmiana kąta żółwia,
- zmiana atrybutu pisaka żółwia,
- zmiana trybu rysowania: "okno"/"płot"/"sklej".

Przerwanie to ma za zadanie uaktywnienie rezydentnego programu użytkownika. Program ten następnie powinien wywołać drugie przerwanie, które zwraca w poszczególnych rejestrach procesora następujące parametry:

- rejestr BX — kąt obrotu żółwia od 0 do 359 stopni,
- rejestr CX — współrzędna X żółwia z zakresu −32000 do 32000,
- rejestr DX — współrzędna Y żółwia z zakresu −32000 do 32000,
- rejestr AL — na bitach D0 do D3 tryb pisaka żółwia:
  - 0 — pisak podniesiony,
  - 1 — pisak opuszczony,
  - 2 — pisak maże poprzedni ślad,
  - 3 — pisak odwraca poprzedni ślad;
- rejestr AL — na bitach D4 do D7 numer koloru pisaka,
- rejestr AH — tryb rysowania żółwia:

0 — "okno",
1 — "sklej",
2 — "płot".

Numery wykorzystywanych przerwań ustala użytkownik w pliku konfiguracyjnym. Niestety, w zasadzie wszystkie przerwania programowe w komputerze IBM-PC są lub potencjalnie mogą być wykorzystane — trudno wskazać najbardziej optymalne numery przerwań. Wydaje się, że najwięcej szans na poprawne działanie rokują przerwania o numerach większych niż #68. Numery przerwań można ustalić za pomocą polecenia IR w pliku konfiguracyjnym (ze względu na potencjalną możliwość niepoprawnej pracy komputera po wykonaniu polecenia IR nie jest ono opisane w Dodatku F) o następującej składni:

IR numer_przerwania_1 numer_przerwania_2

Na przykład, polecenie pliku konfiguracyjnego

IR 108 109

ustala numer przerwania — zgłoszenia zmiany parametrów żółwia na #6C, a numer przerwania — pobrania parametrów żółwia na #6D. W przypadku braku polecenia IR w pliku konfiguracyjnym, przerwania śledzenia ruchu żółwia nie są w ogóle uaktywniane. Informacja o uaktywnieniu przerwań śledzenia ruchu żółwia jest drukowana na ekranie w momencie uruchomienia systemu LOGO — jest to komunikat

Uaktywniono przerwania użytkownika o numerach #NN i #MM,

gdzie NN oznacza szesnastkowy numer przerwania pierwszego, a MM — przerwania drugiego użytkownika.

Opisany mechanizm może być na przykład użyty do naśladowania ruchów żółwia przez dołączone do komputera urządzenia zewnętrzne, takie jak ploter lub poruszający się robot-żółw. Użytkownik powinien napisać program rezydentny, który uaktywniany przez zmianę pozycji żółwia lub sposobu rysowania, tłumaczyłby pobrane współrzędne żółwia i stan pisaka na odpowiednie polecenia dla dołączonego urządzenia. Takie wykorzystanie LOGO może bardzo podnieść walory dydaktyczne tego języka.

Literatura

1. Abelson H.: *Apple Logo*. Mc Graw-Hill, 1982
2. Cellary W., Krysztofiak P., Pielesiak K.: *System operacyjny CP/J i kompilator Junior LOGO*. Materiały konferencji wizytatorów ds. informatyki, Poznań 1988
3. Cellary W., Pielesiak K.: *Leksykon LOGO*. WNT, Warszawa 1990
4. Cellary W., Rykowski J.: *System operacyjny CP/J dla mikrokomputera Elwro 800 Junior*. WNT, Warszawa 1988
5. Gerhold G., Kheriaty L.: *Radio Shack Color LOGO*. Micropi 1982
6. Gilmour J., Hall C., Phipps E.: *CP/M Plus for the PCW8256 and PCW8512 including Dr. Logo*. Amsoft 1985
7. Goldenberg E. P.: *Logo — a cultural glossary. Byte 7 no.* 8 Aug 1982, 210-228
8. Gurbiel E., Krupicka H., Ploski Z.: *Programowanie i LOGO*. Skrypt uczelniany Uniwersytetu Wrocławskiego, Wrocław 1987
9. Pappert S.: Mindstorms: *Children, Computers, and Powerfull Ideas*. Basic Books 1980
10. Williams G.: *Logo for the Apple II, the TI-99/4A, and the TRS-80 Colour Computer. Byte 7 no.* 8 Aug 1982, 230-290
11. *MS-DOS operating system version 5.0 — User's Manual*. Microsoft, 1991

ISBN 83-903197-0-05